최 종
연구보고서

도매시장 상장 농산물 정보의 출하자 입력 시스템 개발

The development of forwarder input system for listed agricultural information of wholesale market

연구기관
조 선 대 학 교

농 림 부

제 출 문

농림부 장관 귀하

본 보고서를 “도매시장 상장 농산물 정보의 출하자 입력시스템 개발” 과제의 최종보고서로 제출합니다.

2006 년 4 월 24 일

주관연구기관명 : 조선대학교

총괄연구책임자 : 최 한 수

세부연구책임자 : 김 영 동

연 구 원 : 이 경 웅

요 약 문

Ⅰ. 제 목

도매시장 상장 농산물 정보의 출하자 입력시스템 개발

Ⅱ. 연구개발의 목적 및 필요성

농산물의 유통경로는 일반적으로 산지, 도매시장, 도매인, 소매인, 소비자로 이어지는 과정을 거치게 된다. 따라서 대부분의 농산물은 도매시장을 경유하게 되며, 도매시장에서는 농산물을 경매하여 중 도매인을 통해 유통시킨다.

도매시장의 업무는 입하, 경매, 경매 후 정산 부분으로 대별할 수 있다. 도매법인에 입하된 농산물의 정보를 법인의 직원이 컴퓨터에 입력하여 상장시키고, 상장된 농산물은 전자식 경매시스템에 의해 경매되어지며, 경매 결과에 따라 정산하는 일련의 전산 시스템이 구축되어 있어서 경매과정의 거의 대부분이 전산화가 이루어져 있다 할 수 있다. 그러나 입하된 농산물을 상장하기 위해 해당 농산물의 정보를 컴퓨터에 입력하는 과정이 너무 많은 시간과 과중한 업무를 요구한다는 것이다. 하나의 도매법인에서 하루에 처리되는 경매 건수가 수백 내지 수천 건에 달하므로 상장을 위해 정보를 입력하는 시간과 처리해야 할 업무량이 너무 과다하다.

본 연구에서는 산지에서 출하자가 자신의 농산물에 대한 정보를 입력하는 시스템을 개발함으로써 농산물을 도매시장에 모아 놓고 도매법인에서 일괄하여 정보를 입력시키는 방법이 갖는 문제점을 해결한다.

컴퓨터를 이용하여 상장 신청서에 수록할 정보들을 도매법인의 컴퓨터로 전송하는 시스템과, 모바일을 이용하여 유무선 통신망으로 상장 정보를 전송하는 시스템을 개발한다.

상장정보를 미리 전송 입력하고 농산물을 도매시장에 보내면 해당 도매법인에서는 농산물을 확인한 후 출하자가 입력한 정보를 바탕으로 상장번호를 부여하

면 입하 농산물의 상장 절차가 완료된다. 따라서 도매법인에서 상장을 위해 농산물 정보를 입력하는 시간과 업무량을 줄이는 효과를 얻을 수 있다.

정보를 입력하는 방법으로는 산지에서 컴퓨터를 사용할 수 있는 입장인 경우에는 컴퓨터를 사용하고, 그렇지 못한 경우에는 유무선 통신망을 이용하는 모바일을 사용할 수 있도록 시스템을 개발한다.

Ⅲ. 연구개발 내용 및 범위

본 연구의 목표는 산지에서 출하자가 자신의 농산물에 대한 상장 정보를 입력하는 시스템을 개발하는 것이다. 컴퓨터를 이용하여 상장 신청서에 수록할 상장 정보들을 도매법인의 서버로 전송하는 시스템과, 모바일을 이용하여 무선 통신망으로 상장 정보를 도매법인의 서버에 전송하는 시스템을 개발하고자 한다.

본 연구에서는 서버 시스템을 구축하고 사용자는 컴퓨터 또는 모바일을 이용하여 어디에서나 작업 할 수 있는 환경 구축을 목표로 하고 있다. 본 과제의 목표 달성을 위한 연구내용은 다음과 같다.

1. 업무 분석

업무 전문가와의 면담을 통하여 전체적인 업무를 파악하고 요구사항을 추출하여 그에 대한 정의를 내린다.

2. DBMS 구축

데이터베이스 관리 시스템을 구축하고 업무 분석 내용을 토대로 객체를 추출하여 추출되어진 객체간의 관계를 정의한 후 테이블을 구축한다.

3. 서버 시스템 설계 및 구축

시스템에 대한 전체적인 구조를 설계한 후 Web Server 및 Web Application Server를 구축하고 테스트를 수행한다.

4. 서버 프로그램 설계 및 구현

업무분석내용에 따라 프로그램의 구조를 설계하고 그 구조에 따라 Webservice를 통해 구현 되어질 객체를 추출하여 구현하고 객체간의 관계를 정의한다.

5. Web Browser 기반의 클라이언트 시스템 구축

사용자 화면을 설계하고 서버에서 구현되어진 WebService의 객체를 정의한 WSDL을 통하여 클라이언트 객체를 구현한 후 전체적인 사용자 인터페이스 프로그램을 구현한다.

6. Mobile 기반의 클라이언트 시스템 구축

PDA 기반의 사용자 시스템의 구현을 위해 Webservice의 객체를 정의한 WSDL을 통하여 모바일 기반의 클래스를 구현하고 Application Program을 통한 사용자 인터페이스 프로그램을 개발한다.

7. 시작품 제작

시작품을 제작한다.

Ⅳ. 연구개발 결과 및 활용에 대한 건의

1. 현행 방법 및 문제점

가. 하나의 도매법인에서 하루에 처리되는 경매 건수가 수백 내지 수천 건에 달한다.

나. 현재 수행하고 있는 일반적인 방법은 출하자가 도매법인에 농산물을 보내오면 해당 농산물의 정보를 접수증에 수기로 작성하고 그 접수증에 작성된 내용을 다시 컴퓨터에 입력하여 상장하는 방법을 채택하고 있다.

다. 도매법인에 입하된 농산물을 상장하기 위해 해당 농산물의 정보를 컴퓨터에 입력하는 과정이 너무 많은 시간과 과중한 업무를 요구한다.

라. 접수증에 농산물의 정보를 작성할 때 수기를 사용함으로써 기록 오류를 발생할 소지를 내포하고 있다.

2. 연구 개발결과

가. 본 연구에서는 산지에서 출하자가 도매법인의 서버에 자신의 농산물에 대한 상장 정보를 입력하는 시스템을 개발하였다.

나. 입력 방법은 컴퓨터를 이용하는 방법, 모바일 기기를 이용하는 방법을 구현하였다.

다. 컴퓨터나 모바일 기기를 출하자가 보유하고 있지 않는 경우에는 계통 출하처인 단위농협이나 영농법인의 컴퓨터를 이용하는 방법이 있다.

3. 활용에 대한 건의

가. 정보화 인프라가 잘 구축된 정보화 선진국으로서 강점을 활용하여 도매법인에서 상장을 위해 농산물 정보를 입력하는 시간과 업무량을 대폭 줄이는 효과를 얻을 수 있어 도매법인의 생산성을 높일 수 있는 방법이다.

나. 상장 정보를 컴퓨터 활자로 출하자가 직접 입력하고 도매법인에서 그 내용을 확인하여 상장하는 방법이므로 출하자와 도매법인 간에 기록 오류로 발생하는 분쟁의 시비를 없앨 수 있는 방법이다.

다. 현행 도매시장의 운영 방법에서 본 시스템을 사용하기 위해 개선해야 할 부분을 보완하면 일정 규모를 갖춘 농가를 대상으로 본 방식이 채택될 수 있을 것으로 전망된다.

SUMMARY

(영문요약문)

A route of the agricultural products distribution is generally a place of product, a wholesale market, a wholesale dealer, a retailer and a customer. Therefore, most of agricultural products are passed through a wholesale market. After an auction, those are distributed by a wholesale dealer at wholesale market.

It is a wholesale market's business that is divided broadly into three category: receiving, auction, account. A wholesale corporation's staff inputs information of received products in the corporation to make list, and those products are sold via auction based on computerization. According to a result of the auction, a process of the auction is almost calculated by computer. However, It takes a long time and a lot of work to input information of the products into a computer for list at a wholesale corporation. It is overwork that one wholesale corporation usually deals with hundreds or thousands of events.

This research is what to develop the system that forwarder input information of agricultural products at the place of product. The system can solve the problem about the things to register information of agricultural products with a wholesale market.

This research develops the system that transmits agricultural products information to a wholesale market server computer by a personal computer and a mobile device. A forwarder inputs list information in advance and delivers agricultural products to a wholesale market. At the wholesale corporation, the staff checks the products and then put grade and list number.

It can reduce a period of registration of agricultural products and events of work for list at a wholesale corporation.

CONTENTS

(영 문 목 차)

목 차

제 1 장 연구개발과제의 개요

제 1 절 연구개발의 목적 및 필요성

농산물의 유통경로는 일반적으로 산지, 도매시장, 도매인, 소매인, 소비자로 이어지는 과정을 거치게 된다. 따라서 대부분의 농산물은 도매시장을 경유하게 되며, 도매시장에서는 농산물을 경매하여 중 도매인을 통해 유통시킨다.

도매시장의 업무는 입하, 경매, 경매 후 정산 부분으로 대별할 수 있다. 도매법인에 입하된 농산물의 정보를 법인의 직원이 컴퓨터에 입력하여 상장시키고, 상장된 농산물은 전자식 경매시스템에 의해 경매되어지며, 경매 결과에 따라 정산하는 일련의 전산 시스템이 구축되어 있어서 경매과정의 거의 대부분이 전산화가 이루어져 있다 할 수 있다. 그러나 입하된 농산물을 상장하기 위해 해당 농산물의 정보를 도매법인에서 컴퓨터에 입력하는 과정이 너무 많은 시간과 과중한 업무를 요구한다는 것이다. 하나의 도매법인에서 하루에 처리되는 경매 건수가 수백 내지 수천 건에 달하므로 상장을 위해 정보를 입력하는 시간과 처리해야 할 업무량이 너무 과다하다.

본 연구에서는 산지에서 출하자가 자신의 농산물에 대한 정보를 입력하는 시스템을 개발함으로써 농산물을 도매시장에 모아 놓고 도매법인에서 일괄하여 정보를 입력시키는 방법이 갖는 문제점을 해결한다.

컴퓨터를 이용하여 상장 신청서에 수록할 정보들을 도매법인의 컴퓨터로 전송하는 시스템과, 모바일을 이용하여 유무선 통신망으로 상장 정보를 전송하는 시스템을 개발한다.

상장정보를 미리 전송 입력하고 농산물을 도매시장에 보내면 해당 도매법인에서는 농산물을 확인한 후 출하자가 입력한 정보를 바탕으로 상장번호를 부여하면 입하 농산물의 상장 절차가 완료된다. 따라서 도매법인에서 상장을 위해 농산물 정보를 입력하는 시간과 업무량을 줄이는 효과를 얻을 수 있다.

정보를 입력하는 방법으로는 산지에서 컴퓨터를 사용할 수 있는 입장인 경우에는 컴퓨터를 사용하고, 그렇지 못한 경우에는 유무선 통신망을 이용하는 모바일을 사용할 수 있도록 시스템을 개발하였다.

제 2 절 연구개발 내용 및 범위

1. 연구개발 목표와 내용(2005.4-2006.4)

연 구 개 발 목 표	연구개발 내용 및 범위
○ 업무 분석	· 입하과정 업무 분석 및 업무 정의서 작성 · 요구사항 조사 및 요구사항 정의서 작성
○ DBMS 구축	· DBMS 구축 및 테스트 · 데이터 플로우 다이어그램 작성 · Table 정의 · Table 구축 · 데이터의 무결성 확보
○ 서버 시스템 설계 및 구축	· 전체 시스템 설계 · Web Server 구축 및 테스트 · WAS 구축 및 테스트
○ 서버 프로그램 설계 및 구현	· 사용자 화면 설계 및 디자인 · WebService를 위한 원격객체 정의 및 구현
○ Web Browser 기반의 클라이언트 시스템 구축	· WSDL을 이용한 클래스 구현 · 사용자 화면의 구성 · 사용자 인터페이스 프로그램
○ Mobile 기반의 클라이언트 시스템 구축	· PDA 기반 시스템 설계 · WSDL을 이용한 클래스 구현 · Web Browser를 통한 PDA 사용자 프로그램 구현 · Application Program을 통한 PDA 사용자 프로그램 구현
○ 시작품 제작	· 시작품 제작

본 연구의 목표는 산지에서 출하자가 자신의 농산물에 대한 상장 정보를 입력하는 시스템을 개발하는 것이다. 컴퓨터를 이용하여 상장 신청서에 수록할 상장 정보들을 도매법인의 서버로 전송하는 시스템과, 모바일을 이용하여 유무선 통신망으로 상장 정보를 도매법인의 서버에 전송하는 시스템을 개발하고자 한다.

본 연구에서는 서버 시스템을 구축하고 사용자는 컴퓨터 또는 모바일을 이용하여 어디에서나 작업할 수 있는 환경 구축을 목표로 하고 있다.

제 2 장 국내외 기술개발 현황

제 1 절 국내·외 관련기술의 현황과 문제점

1. 국내기술현황

국내에서는 도매시장에 입하된 농산물에 대한 정보를 접수증에 수기로 작성한 후 접수증에 기재된 내용을 컴퓨터에 입력시키는 방법을 취하고 있다.

이 방법은 하루에 수백 또는 수천 건을 처리해야 하는 도매법인에서 업무량이 너무 과다하다.

전자식경매 장치를 이용해 경매를 진행하는 도매법인의 경우 경매를 위해 도매법인의 서버에 정보를 입력함으로써 상장이 되는데 입하량이 너무 많은 날에는 미쳐 농산물의 정보를 입력할 수 없어 당일 입하된 농산물의 전부를 전자식으로 경매하지 못하고 일부를 수지식으로 경매를 해야 할 경우가 발생한다.

도매시장에서 접수증에 상장할 농산물의 정보 즉 품목, 품종, 수량, 등급 등을 수기로 기록하고 접수증에 수기로 기록된 내용을 다시 컴퓨터에 입력하는 과정에서 오류가 발생할 가능성이 있어 도매법인과 출하자 간에 분쟁의 소지를 내포하고 있다.

2. 국외기술현황

네델란드의 화훼 도매시장의 경우 바코드를 이용하여 상장 정보를 자동 입력시키는 방법을 채용하고 있다.

바코드 방식은 산지에서 농산물의 상장정보를 바코드로 표현하고 출력하여 농산물에 바코드를 부착한 상태로 농산물을 도매시장으로 보내면 도매시장에서는 바코드 리더(reader)로 정보를 읽히고 그 정보를 컴퓨터에 전송하는 시스템을 채용하고 있다.

제 3 장　연구개발수행 내용 및 결과

1. 업무 분석

“농산물 상장 등록 시스템”의 개발을 위해 먼저 전체적인 업무 정의를 위해 현장 업무 전문가와의 면담을 통해 업무를 이해하고 업무 전문가의 요구 사항을 정의하였다.

가. 입하과정 업무 분석 및 업무 정의서 작성

업무 전문가와의 면담을 위해 광주광역시 소재 각화동 농산물 공판장의 전산담당자와 면담을 실시하였다.

1) 업무 전문가와 면담 내용

업무 전문가와의 면담 내용을 다음과 같이 정리하였다.

가) 출하자 정보에 주민등록 번호가 기재되어야 한다(2003년 국세청 권고사항).

나) 출하는 개인이 하는 개인 출하와 농협을 통하는 계통 출하로 구분된다.

다) 품목, 품종, 중량, 수량, 등급 등의 정보가 필요하며 과일의 경우 과수정보가 필요하다.

라) 하역은 하역 노조에서 담당하며 경매 시작 전까지 입하되어진 농산물에 대한 관리를 한다.

마) 출하자의 경우 18시 까지 수확이 이루어지며 20시경 상차하고 공판장에는 22시~02시 경에 도착한다.

바) 등급은 출하자가 기재하며 변경은 거의 일어나지 않는다. 그러나 경매 시에는 경매사의 판단에 의해 변경되어지기도 한다.

사) 본 시스템의 활용은 출하자의 의지에 달려 있다.

아) 품목, 품종, 등급, 산지 코드는 농림부 홈페이지를 참조하기 바란다.

자) 물량 조정을 위해 당일 경매 대상에 대한 전체적인 통계자료가 있으면 용이하겠다.

차) 기존의 수탁판매 접수증을 참조하여 구성하면 용이 하겠다.

2) 시스템의 주요 구성

업무 담당자와의 면담을 통하여 시스템의 구성에 필요한 내용을 점검하고 이를 통하여 시스템을 구성한다.

가) 출하자 정보에 주민등록 번호, 은행 계좌번호 등을 반드시 기입하게 한다. 시스템에서 한번 등록이 되면 매번 입력하지 않아도 이용할 수 있도록 한다.

나) 품목, 품종, 등급, 산지 코드 등은 농림부에서 지정한 코드를 사용한다.

다) 시스템의 사용은 출하자, 접수자, 관리자 등으로 구분 관리한다.

라) PDA를 통한 시스템 접근이 Internet Browser를 통한 접근과 그 역할이 동일해야 한다.

마) 입하되어진 농산물에 대해서는 변경이 불가하다.

바) 하역비는 계산식에 의해 결정되어진다.

사) 접수증의 삭제는 관리자만 가능하다.

아) 접수자에 의한 농산물 내용 변경은 로그를 남기어 추후 분쟁을 없앤다.

자) 접수증의 상태는 '입하 준비중', '입하', '삭제'로 구분한다.

차) 시스템에 대한 접근은 시간적 공간적 제약을 최소화 하도록 한다.

나. 요구사항 조사 및 요구사항 정의서 작성

1) 면담 내용 분석

면담 내용 분석을 기초로 농산물 상장 등록 시스템의 구현 내용을 정의하고 이를 기준으로 시스템을 구현하고자 한다. 면담 내용을 토대로 출하자, 접수자, 관리자 각각의 Use Case를 정의하였다. 본 시스템의 주요 대상은 다음과 같다.

- 출하자는 농민 및 농협직원이 된다.
- 접수자는 공판장에서 하역을 담당하는 직원이 된다.
- 관리자는 공판장의 전산담당자가 된다.

각 대상별 역할은 다음과 같다.

가) 출하자

출하자는 통계 데이터를 통해 당일 접수 사항을 조회할 수 있으며 이를 통하여 출하시기를 조정할 수 있다.

- 사용자 등록을 할 수 있다.
- 수탁판매 농산물을 접수할 수 있다.
- 접수되어진 농산물을 조회할 수 있다.

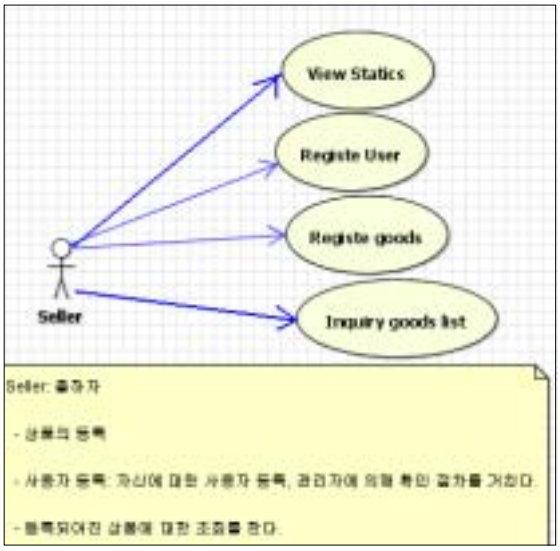

그림 3.1.1 출하자 Use Case Diagram

나) 접수자

접수자는 입하 준비중인 접수 건에 대해 실제 물량과 접수되어진 농산물간의 비교를 통해 농산물의 입하를 확인한다.

- 통계 데이터를 통해 입하 대기 중인 농산물에 대한 정보를 알 수 있다.
- 사용자 등록을 할 수 있다.
- 입하 준비 중인 농산물 리스트를 확인할 수 있다.
- 시스템에 등록되어진 농산물 내역과 입하 준비 중인 농산물 내역이 상이 할 경우 농산물 정보를 변경할 수 있다.

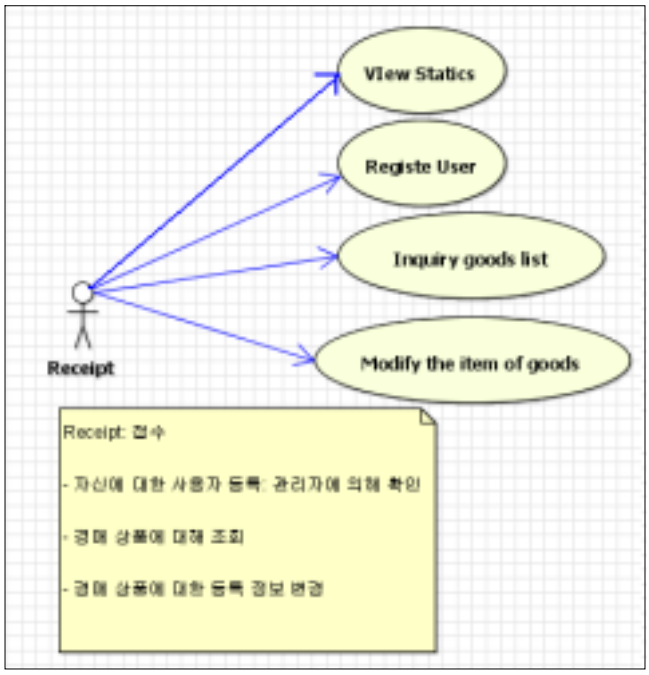

그림 3.1.2 접수자 Use Case Diagram

다) 관리자

관리자는 시스템의 전반적인 관리를 담당한다.

- 통계 데이터를 통해 당일 경매 물량을 예측할 수 있으며 또한 물량 조정을 할 수 있다.
- 시스템 사용 요청자에 대한 관리를 한다.
- 시스템에 접수되어진 접수증 중 아직 입하 확인되어지지 않은 건에 대한 삭제를 할 수 있다.
- 시스템 운영에 관련된 코드를 관리한다.

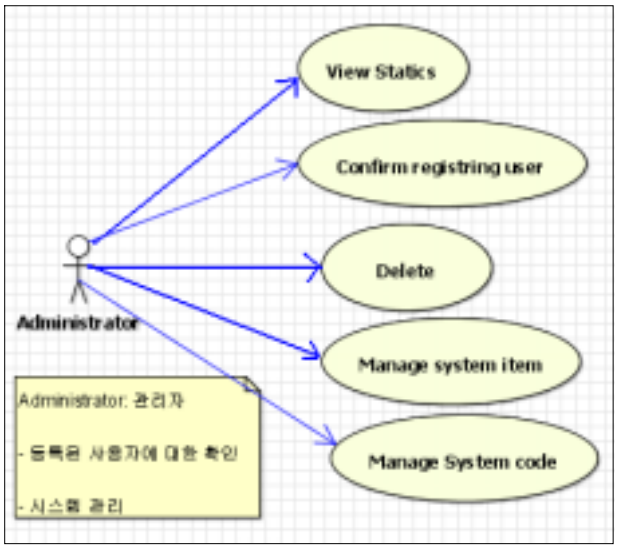

그림 3.1.3 관리자 Use Case Diagram

2) 농산물 상장 등록 시스템 구성에 대한 기술

농산물 상장 등록 시스템의 일반적인 사항을 기술하고 이를 통하여 시스템을 구성한다.

가) 농산물 상장 등록 시스템의 사용자는 크게 농산물을 출하하는 출하자와 출하된 농산물을 공판장 등에서 하역을 하고 입하를 받는 접수자 그리고 시스템의 전반적인 관리를 하는 관리자로 나눌 수 있다.

나) 출하는 일반적으로 경매시행일 전일 저녁과 당일 새벽에 이루어진다. 따라서 하역 작업은 주로 당일 새벽에 이루어진다고 볼 수 있다.

다) 출하 시 출하자는 농산물에 대한 출하 정보를 저장하고 수송 담당자에게 관리 번호 또는 출하자의 시스템 사용 ID를 알려 준다.

라) 접수자는 입하되어진 농산물에 대해 농산물 내역에 대해서 변경이 가능하나 여타 정보에 대해서는 변경이 불가능 하다. 또한 변경되어진 농산물내역은 추후 출하자와 접수자 간의 분쟁을 방지 하기위해 별도로 관리 한다.

마) 출하자가 농산물을 접수하면 하역비는 자동 계산되어지도록 한다.

바) 접수자에 의해 입하 확인되어진 접수 건에 대해서는 더 이상 변경이 불가능하다.

사) 시스템의 사용을 위해서는 먼저 등록 요청을 하고 관리자는 해당 요청자의

직접 확인을 통해 사용허가를 해준다.

아) 관리자는 이미 접수되어진 건에 대해 삭제가 가능하다. 접수 건에 대한 삭제는 관리자만이 행할 수 있다.

자) 전체적인 등록 건에 대한 통계를 통하여 출하자의 경우 당일 출하 여부를 결정하고 관리자의 경우 전체적인 물량을 예상할 수 있도록 통계 자료를 보여준다.

차) 시스템에서 사용되어지는 코드는 농림부 홈페이지에서 제공하는 코드를 사용하며 정확한 정보를 유지할 수 있도록 한다.

3) 전체 업무 흐름도

농산물 상장 등록 시스템 은 크게 출하자, 접수자, 관리자로 분류되며 출하자는 실제 수탁판매품을 등록하는 농민 및 농협직원 등 다양한 사람이 대상이 될 수 있으며 접수자는 공판장의 하역을 담당하는 사람으로 접수증에 등록된 물품과 실제 입하되어지는 물품이 동일한지를 비교하고 수탁판매품의 입하를 확인한다. 관리자는 농산물 상장 등록 시스템의 전반적인 사항을 관리한다. 또한 관리자는 등록되어진 수탁판매품 등록증을 삭제할 수도 있다.

농산물 상장 등록 시스템 은 접수 번호를 기본 키 값으로 활용한다. 따라서 모든 등록 접수 삭제 수정 등이 접수번호에 의해 조정되어진다. 또한 출하자, 접수자, 관리자의 권한을 각각 부여하여 동일 시스템에 대해 다른 메뉴 환경을 제공하였다.

접수는 크게 '입하 준비중'과 '입하'로 구분되어지는데 입하 준비중인 접수에 대해서는 수정 삭제가 가능하나 입하가 확인되어진 접수에 대해서는 수정 삭제가 불가하게 된다. 또한 접수자에 의한 접수 내용의 변경은 변경 내용을 Server에 저장하여 분쟁의 여지를 최소화 하였다.

본 시스템의 주요 관리 항목으로 접수 번호와 관리 번호가 있다. 접수 번호는 출하자가 등록 요청만 있어도 무조건 증가하는 번호이고 관리 번호는 실제 등록이 발생 할 경우에 증가 하는 번호이다. 따라서 관리번호를 통해서는 당월 등록 건수를 파악 할 수 있게 된다. 그리고 접수 번호는 농산물 상장 등록 시스템 전체의 주요 구분 값으로 이용되어진다.

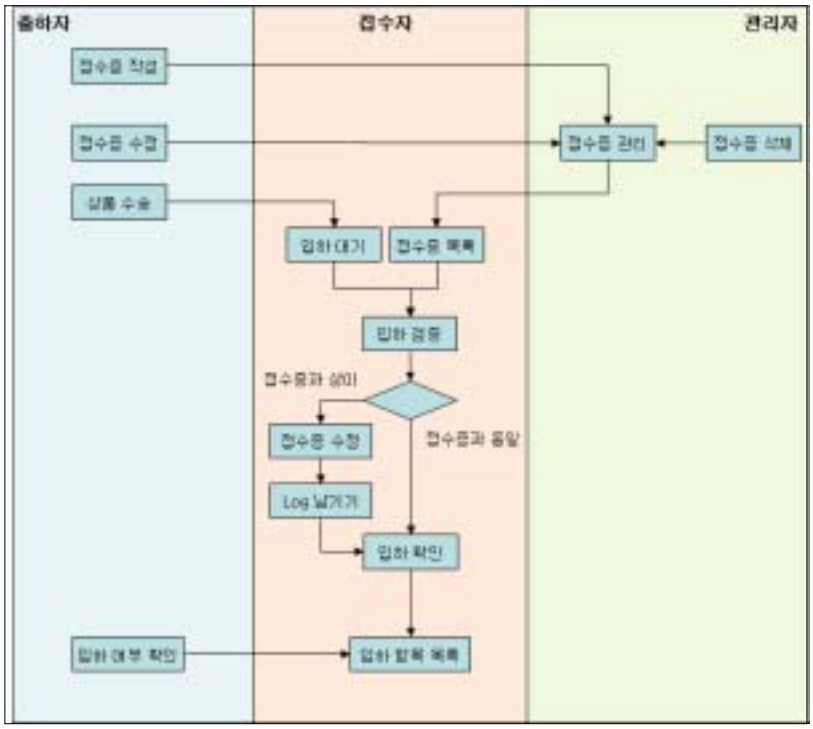

그림 3.1.4 전체 업무 흐름도

4) 출하자 농산물 등록 업무 흐름도

시스템에 login을 요청하게 되고 등록된 사용자가 아닐 경우 login 화면으로 이동한다. 등록된 사용자의 경우 권한에 따라 사용 가능 메뉴를 보여준다. 농산물 등록 업무는 출하자만이 할 수 있으며 출하자가 접수 등록 요청을 하면 접수 번호를 부여 받는다. 접수 번호는 접수 요청이 있으면 무조건 발급되며 실제 접수가 안돼도 보존된다. 접수증을 작성하고 접수 요청을 하게 되면 관리번호가 부여되고 Data base에 해당 내용이 저장되어진다. 관리 번호는 접수 번호와 같이 계속 증가되는 숫자로 접수 번호는 접수 여부와 상관없이 증가하는 반면에 관리번호는 실제 접수가 일어나야 증가한다. 따라서 관리 번호를 보면 당월 접수 건수를 알 수 있다. 관리 번호는 '연(4자리)+월(2자리)+일련번호'로 구성되어진다.

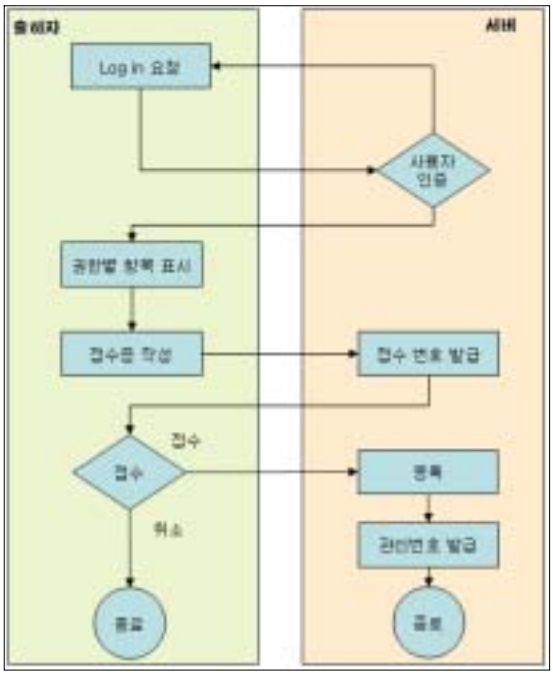

그림 3.1.5 출하자 농산물 등록 업무 흐름도

5) 접수 관리

시스템에 login을 요청하게 되고 등록된 사용자가 아닐 경우 login 화면으로 이동한다. 등록된 사용자의 경우 권한에 따라 사용 가능 메뉴를 보여준다. 접수 관리는 접수자에 의해 행하여지는데 접수자는 출하자가 등록한 접수증의 내용과 실제 하역되어진 농산물과 내용이 동일한지를 확인하고 입하 여부를 결정한다. 접수자는 실제 하역 내용과 접수증의 내용이 상이할 경우 변경할 수 있으며 변경 내용은 Server에 저장 관리되어지므로 추후의 분쟁을 방지할 수 있다.

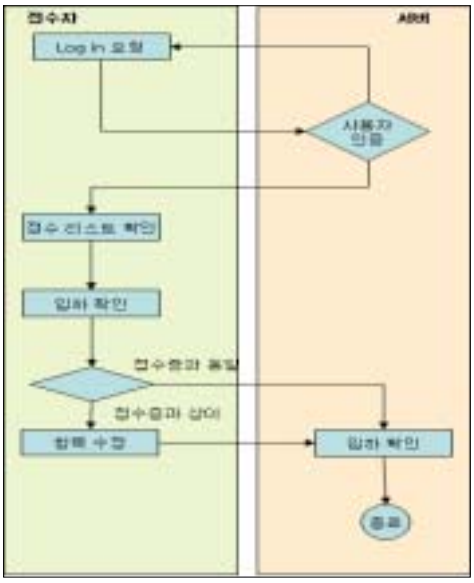

그림 3.1.6 접수 관리 업무 흐름도

6) 사용자 관리

시스템을 사용하고자 하는 사용자는 우선 사용자 등록 요청을 하게 되고 관리자는 사용자 등록 요청 리스트를 확인하고 요청한 사용자와 직접 연락하여 사용 여부 및 권한 부여 여부를 결정한다.

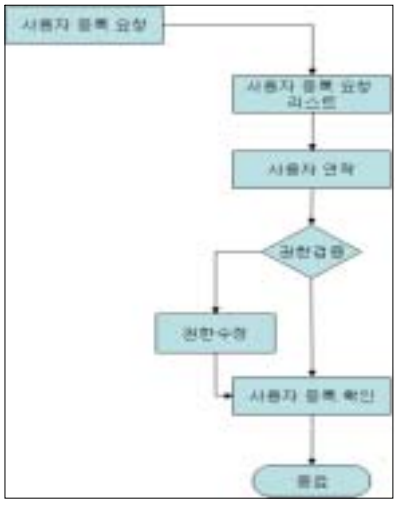

그림 3.1.7 사용자 관리 업무
흐름도

7) 접수 삭제

시스템에 login을 요청하게 되고 등록된 사용자가 아닐 경우 login 화면으로 이동한다. 등록된 사용자의 경우 권한에 따라 사용 가능 메뉴를 보여준다. 접수 삭제는 관리자에 의해서만 이루어지는데 이미 입하 확인이 된 접수에 대해서는 삭제 할 수 없다. 관리자는 login한 후 접수 리스트를 확인하고 입하 확인이 안 된 경우는 해당 접수를 삭제한다.

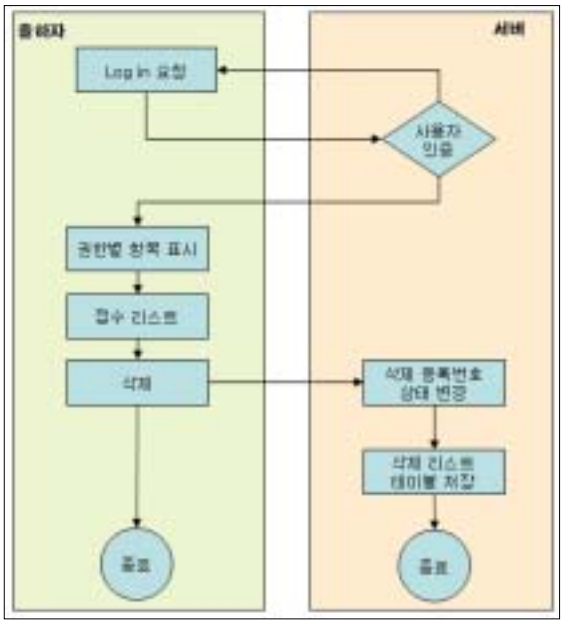

그림 3.1.8 접수 삭제 업무 흐름도

8) 접수 수정

시스템에 login을 요청하게 되고 등록된 사용자가 아닐 경우 login 화면으로 이동한다. 등록된 사용자의 경우 권한에 따라 사용 가능 메뉴를 보여준다. 출하자는 자신의 접수증에 대한 수정이 가능하다.

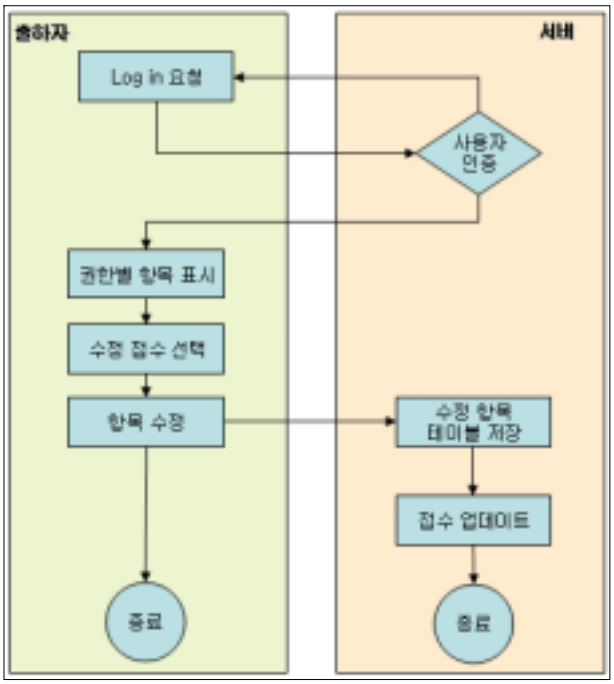

그림 3.1.9 접수 수정 업무 흐름도

2. DBMS 구축

본 연구에서는 소규모 시스템에 적절한 Windows 기반의 Mysql DBMS를 사용하여 서버를 구축하였으며 SQL 작성은 표준 PL-SQL로 작성하여 DBMS에 종속적이지 않도록 구성하였다.

가. DBMS 구축 및 테스트

DBMS(DataBase Management System)는 Open source기반의 Mysql 5.0을 사용하였다. 기존 버전에 비해 기능이 향상되고 특히 stored procedure와 같은 기능이 추가되었으며 다른 DBMS에 비해 낮은 서버 성능을 요하며 또한 다양한 OS(Operating System) 환경에 설치가 가능하여 대규모 시스템이 아닌 중소형 시스템에서의 사용에 적절하다.

1) Mysql

MySQL은 멀티유저, 멀티 쓰레드 SQL(Structured Query Language)데이타베

이스 서버로써 DB서버 데몬인 mysqld 와 여러 가지 이용자 프로그램 그리고 라이브러리로 구성되어져 있다. MySQL의 주 목표는 속도, 뛰어난 수행능력 그리고 사용의 편리함이며 애초에 MySQL은 어떤 제조업체가 제공하는 SQL 서버 보다 강력한 DB서버가 필요한 TcX 내부에서 자체적으로 사용할 목적으로 만들어 졌다. TcX에서는 MySQL을 1996부터 사용해 왔는데 현재 700만 레코드 이상 되는 500개 이상의 테이블이 (100 기가 바이트 이상) 주요 업무에 사용되고 있다. MySQL은 높은 수행능력을 발휘하는 것을 목표로 수년 전 개발이 시작되었으며 여전히 개발이 진행 중이지만 매우 풍부하고 유용한 함수들을 제공한다.

2) Mysql의 특징

가) 커널 스레드를 이용한 멀티 스레드를 지원한다. 따라서, CPU가 여러 개 있을 경우 여러 개의 CPU를 잘 활용할 수 있다.

나) C, C++, Java, Perl, PHP, Python과 TCL에 대한 API를 제공한다.

다) 여러 플랫폼에서 작동한다.

라) 많은 컬럼 타입을 제공한다.

마) one-sweep multijoin을 이용하여 Join이 매우 빠르다.

바) SQL 함수들은 Optimized된 class library를 이용하여 구현되었다. 또한 매우 빠르며, 불필요한 메모리 할당을 하지 않는다.

사) 권한과 암호 시스템은 flexible하고 보안이 잘 되어 있다. 원격의 서버에 접속할 때 모든 암호들은 암호화 되어 전송된다.

아) ODBC를 지원한다. 따라서 Access를 이용해서 MySQL에 연결할 수 있다.

자) 테이블에 16개의 인덱스를 줄 수 있다.

차) 크기가 큰 데이타 베이스도 다룰 수 있다. 50,000,000 개의 레코드를 가지고 있는 데이타베이스도 다루고 있다.

카) 메모리 누수가 없다.

타) isamchk라는 유틸리티로 테이블 검사, 성능향상, 수리 등을 할 수 있다.

3) MySQL의 설치

MySQL Server는 MySQL Web site에서 다운 받아 설치 하게 된다. 본 연구에서는 'mysql-5.0.19-win32'을 설치하였다. 그리고 MySQL의 관리를 위하여 'mysql

-administrator-1.1.9-win.msi'을 다운 받아 설치하였으며 query 테스트 등을 위하여 MySQL Web Site에서 제공하는 'mysql-query-browser-1.1.20-win.msi'을 설치하였다.

가) MySQL Server의 설치

MySQL Server를 설치 시에 서버가 Test 서버인지 운영서버인지를 결정하고 통신 포트 및 기본 데이터베이스 타입, 관리자 계정과 비밀번호 등을 정의하도록 되어 있다.

그림 3.2.1 MySQL 서버 설치 화면

나) MySQL Administrator 및 Mysql Query Browser

MySQL Administrator는 MySQL 서버가 설치된 후에 설치하여야 하며 설치 시 서버의 정보를 입력해 주어야 한다. Mysql Query Browser는 설치 후 실행 시에 서버의 정보를 입력하면 된다.

그림 3.2.2 MySQL Administrator 설치 화면

그림 3.2.3 MySQL Query Browser 설치 화면

다) Schema 설치

설치 된 서버에 본 연구에서 사용할 Schema를 설치한다. Schema는 ACAI로 명명하며 MySQL Administrator를 이용하여 설치하였다. Schema가 설치된 후에 각각의 Table을 구성할 수 있다.

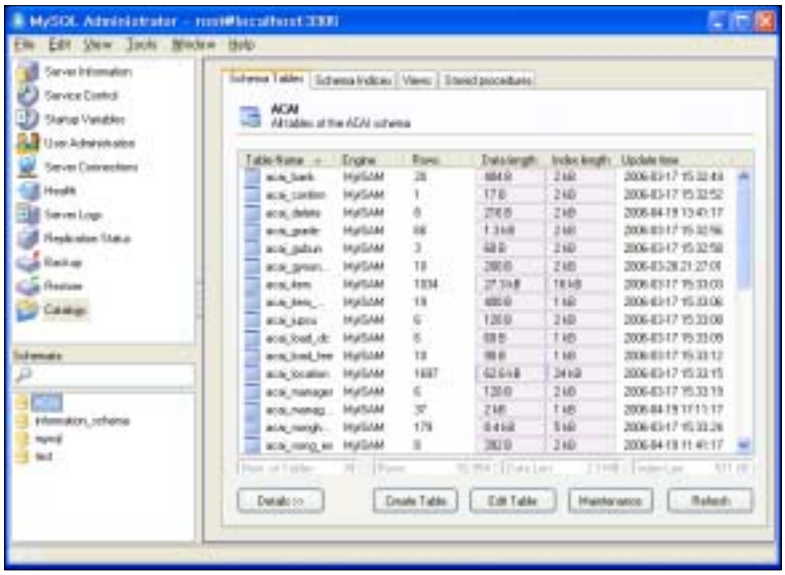

그림 3.2.4 Administrator를 이용한 Schema 설치

4) 테스트

가) MySQL 서버의 정상적인 작동을 확인 하기위해서 mysql/bin/mysqld.exe 파일을 실행하여 정상적인 실행이 이루어지는 가를 확인하였다.

그림 3.2.5 MySQL 서버 테스트 화면

나) MySQL Administrator의 테스트를 위해 MySQL 서버에 접속 연결 테스트 및 MySQL 기본 Schema인 mysql Schema의 테이블을 확인하였다.

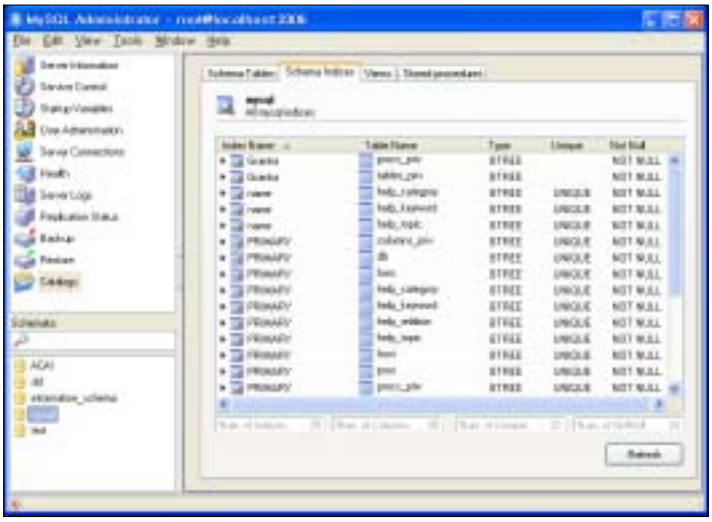

그림 3.2.6 MySQL Administrator 설치 화면

다) Mysql Query Browser

Query Browser의 테스트는 크게 서버의 접속여부와 query의 정상적인 실행을 들 수 있다. 기 설치된 MySQL 서버에 접속하여 mysql Schema의 proc 테이블에 대해 query를 실행하고 정상적인 작동을 확인하였다.

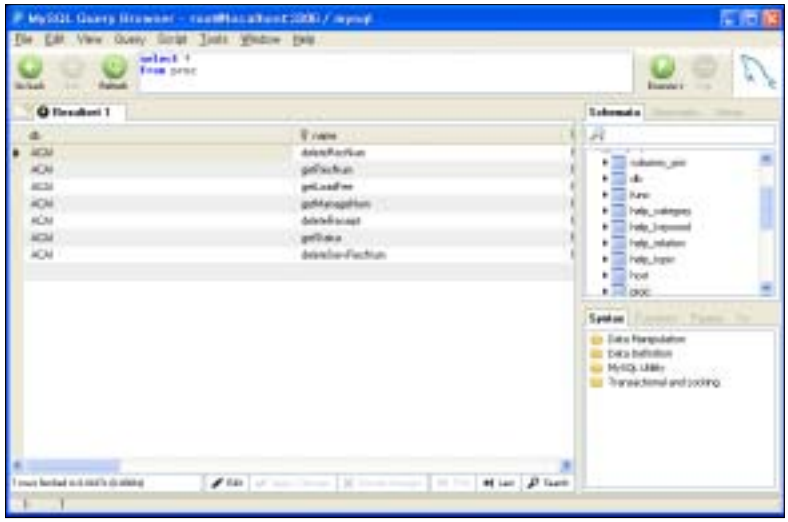

그림 3.2.7 MySQL Query Browser 테스트 화면

나. 데이터 플로우 다이어그램 작성

위에서 정의되어진 테이블을 기반으로 테이블간의 연관성 및 데이터 연관성을 정의하기 위하여 ERD(Entity Relation Diagram)를 작성하였다. 각각의 Foreign Key를 기준으로 연관성을 정의하였으며 별도의 Index를 추가하지는 않았다. 본 시스템의 ERD는 다음 그림과 같다.

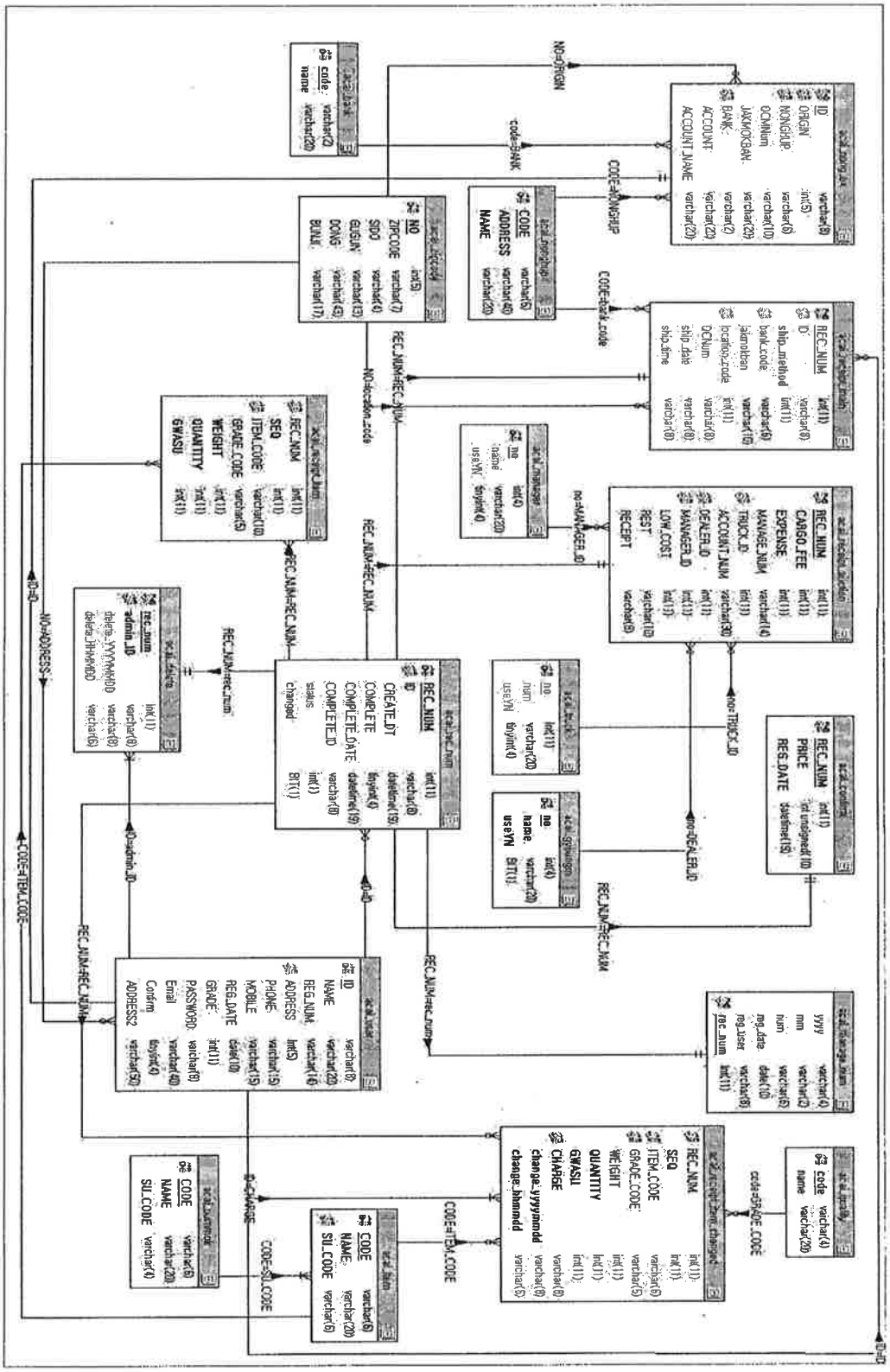

그림 3.2.8 ERD(Entity Relation Diagram)

자료출처 : 도매시장 상장 농산물 정보의 출하자 입력 시스템 개발 / 농림부

다. Table 정의

1) Table Type의 분류

MySQL의 테이블은 크게 TST(Transaction Safe Table)와 NTST(Not Transaction Safe)두 분류로 나눌 수 있다. MySQL은 MyISAM, ISAM, BDB, InnoDB의 5개 테이블 중에서 한 가지를 선택하여 만들 수 있다.

표 3.2.1 테이블 타입 비교

분류	장 점	테이블 타입
TST	* MySQL에 문제가 생길 경우 테이터를 회복 시키기 쉽기 때문에 안전하다. * Update가 실패했을 경우에 이전의 데이터가 자동으로 복구 된다.	MyISAM, ISAM, HEAP
NTST	* 속도가 빠르다. * 디스크 용량을 적게 차지 한다. * 메모리를 적게 사용한다.	BDB, InnoDB

가) MyISAM

MyISAM 테이블 타입은 MySQL 3.23에서의 기본 테이블이다. 3.22이하 버전의 MySQL테이블 타입인 ISAM을 기초로 하여 만들어진 테이블 타입으로 많은 기능이 확장되었다. ISAM에 비해 확장된 기능은 다음과 같다.

- OS에 따라서 4GB이상 데이터를 저장할 수 있다.
- BLOB과 Text도 인덱스로 사용할 수 있다.
- NULL인 컬럼도 인덱스로 사용할 수 있다.
- 키의 길이가 500바이트이다.
- 테이블이 올바르게 닫혔는지 알 수 있다.
- 하나의 테이블 당 32개의 키를 가질 수 있다.

나) ISAM

MySQL 3.22버전 이하에서 쓰이던 테이블 타입이다. MySQL 3.23버전에서는 ISAM, MyISAM, BDB테입르 타입중 원하는 테이블 형태를 골라서 테이블을 만

들 수 있지만 MySQL 3.22버전 이하에서는 ISAM테이블 타입만 쓸 수 있다. ISAM 테이블의 특징은 다음과 같다.

- OS에 종속적이나 속도가 빠르다.
- 테이블의 최대 크기는 4GB이다.
- 최대 키 수가 작다.
- Fragment가 생기기 쉽다.

다) HEAP

테이블의 내용이 메모리에 저장되는 테이블 타입이 HEAP 테이블이다. 대신 휘발성인 메모리에 저장되므로 MySQL이 종료되면 모든 데이터가 사라진다. 하지만 속도가 매우 빠름으로 작업 중에 임시 테이블로 사용하기 좋다. HEAP 테이블의 특징 및 주의 점은 다음과 같다.

- HEAP테이블을 생성할 때는 MAX_ROWS를 이용하여 HEAP테이블이 메모리를 모두 소모하지 않도록 해야 한다.
- HEAP 테이블을 더 이상 사용하지 않을 때는 DROP TABLE 또는 DELETE FROM을 이용하여 테이블이나 레코드를 삭제해 메모리를 반환해야 한다.
- HEAP 테이블에서는 AUTO_INCREMENT나 TEXT/BLOB을 사용할 수 없다.
- HEAP테이블을 생성한 클라이언트 이외의 클라이언트에서도 만들어진 HEAP 테이블을 이용할 수 있다.
- max_heap_table_size보다 큰 테이블을 만들 수 없다.

라) BDB

MySQL은 속도를 빠르게 하기 위해 Transaction을 기본적으로 지원하지 않는다. 하지만 MySQL 3.23.15버전부터는 BDB를 이용하여 기본적인 트랜잭션을 사용할 수 있다. BDB 테이블의 특징은 다음과 같다.

- BDB테이블에는 Primary Key가 꼭 있어야 한다.
- ALTER TABLE을 사용할 수 없다.
- 속도가 느리다.
- 로그파일의 크기가 쉽게 커지므로 자주 로그 파일을 삭제해야 한다.

마) InnoDB

InnoDB는 MySQL에서 최근 지원하기 시작한 테이블 타입으로 많은 기능이 향상된 테이블 타입이다. 현재 유명한 Slashdot.org에서도 1TB이상의 데이터를 InnoDB를 이용하여 저장하고 있다. 또한 InnoDB를 이용하여 초당 800개의 Insert와 Update를 처리하는 사이트도 있다. InnoDB는 Mysql 3.23.34a에서부터 포함되기 시작했다. InnoDB의 특징은 다음과 같다.

- 트랜잭션을 지원한다.
- Row Level Locking과 오라클과 비슷한 Select시 Locking을 안하는 방법을 지원한다.
- MySQL에서 처음으로 Foreign Key를 지원한다.
- 다중 사용자 환경에서 좋은 성능을 발휘한다.
- CPU를 효과적으로 사용한다.
- 많은 데이터도 효율적으로 사용할 수 있다.

바) 본 연구에서는 MyISAM에 비해 데이터 테이블의 직관적 관리는 불편하나 다양한 향상된 성능으로 Transaction 및 Foreign Key 사용이 가능하며 테이블 타입 중 많은 데이터를 효과적으로 사용할 수 있는 InnoDB를 사용하였다.

2) 테이블 정의

본 연구에서는 ACAI라는 Schema를 만들어 테이블을 정의 하였다. 테이블 타입은 InnoDB를 사용하였으며 테이블 간 Primary Key 및 Foreign Key를 정의하여 사용하였다. 또한 Stored Procedure를 작성하여 효율성을 증대 시켰다. 테이블의 기본 정보는 다음과 같다.

표 3.2.2 농산물 상장 등록 시스템의 테이블 정의

Name	Engine	Version	Row_format	Rows	Avg_row_length
acai_bank	InnoDB	10	Compact	20	819
acai_confirm	InnoDB	10	Compact	1	16384

acai_delete	InnoDB	10	Compact	2	8192
acai_grade	InnoDB	10	Compact	66	248
acai_gubun	InnoDB	10	Compact	3	5461
acai_gyoungm	InnoDB	10	Compact	10	1638
acai_item	InnoDB	10	Compact	1097	74
acai_item_main	InnoDB	10	Compact	19	862
acai_jupsu	InnoDB	10	Compact	6	2730
acai_load_dc	InnoDB	10	Compact	6	2730
acai_load_fee	InnoDB	10	Compact	10	1638
acai_location	InnoDB	10	Compact	1615	81
acai_manage_num	InnoDB	10	Compact	67	244
acai_manager	InnoDB	10	Compact	6	2730
acai_nong_ex	InnoDB	10	Compact	5	3276
acai_nonghup	InnoDB	10	Compact	179	91
acai_pummok	InnoDB	10	Compact	385	42
acai_quality	InnoDB	10	Compact	25	655
acai_rec_num	InnoDB	10	Compact	82	199
acai_receipt_auction	InnoDB	10	Compact	19	862
acai_receipt_item	InnoDB	10	Compact	24	682
acai_receipt_item_changed	InnoDB	10	Compact	11	1489
acai_receipt_main	InnoDB	10	Compact	19	862
acai_temp	InnoDB	10	Compact	1227	106
acai_truck	InnoDB	10	Compact	4	4096
acai_unit	InnoDB	10	Compact	19	862
acai_user	InnoDB	10	Compact	7	2340
acai_weight	InnoDB	10	Compact	10	1638
acai_wrap	InnoDB	10	Compact	54	303
acai_zipcode	InnoDB	10	Compact	48963	75
board	InnoDB	10	Compact	1	16384
boardfile	InnoDB	10	Compact	1	16384

commentmng	InnoDB	10	Compact	1	16384
downloadmng	InnoDB	10	Compact	1	16384
downloadmnginfo	InnoDB	10	Compact	1	16384
kkaok	InnoDB	10	Compact	2	8192
menu	InnoDB	10	Compact	15	1092
tablemng	InnoDB	10	Compact	1	16384
tablemnginfo	InnoDB	10	Compact	1	16384

라. 데이터의 무결성 확보

데이터 무결성이란 데이터베이스에 저장된 모든 데이터 값이 정확한 상태로 저장된 것을 의미한다. 정확하지 않은 데이터 값이 데이터베이스에 저장된 것을 데이터 무결성이 위반되었다고 표현한다.

1) 데이터 무결성의 종류

데이터 무결성의 종류는 다음과 같다.

가) 개체 무결성 (Entity integrity)

- 테이블에 있는 모든 행들이 유일한 식별자를 가질 것을 요구한다.

나) 영역 무결성 (Domain intrgrity)

- 한 컬럼에 대해 NULL의 허용 여부와 타당한 데이터 값들을 지정한다.
- 자료형, 규칙과 제약, 값 범위 등을 제한한다.

다) 참조 무결성 (Referential integrity)

- 기본 키와 참조 키 간의 관계가 항상 유지됨을 보장한다.
- 참조되는 테이블의 행을 이를 참조하는 참조키가 존재하는 한 삭제될 수 없고 기본 키도 변경될 수 없다.

2) 데이터 무결성 확보

본 시스템에서는 무결성 확보를 위하여 다음과 같이 시행 하였다.

가) 참조키를 통한 데이터 무결성 확보

참조키를 통하여 데이터들의 연관성을 확보하였다.

나) 기본 키를 통한 데이터 중복 방지

데이터 중복에 따른 무결성 위반을 방지하기 위하여 중복이 허용되지 않은 필드에 대해 기본 키를 설정하였다.

3) 테이블의 키 설정

테이블을 구성하기 전에 각 테이블에 대한 키를 설정하였다. 주요 테이블의 키 설정은 다음과 같다.

가) acai_receipt_auction

- PRIMARY KEY : REC_NUM
- FOREIGN KEY : MANAGER_ID REFERENCES acai_manager(no)
- FOREIGN KEY : REC_NUM REFERENCES acai_rec_num(REC_NUM)
- FOREIGN KEY: TRUCK_ID REFERENCES acai_truck(no)
- FOREIGN KEY: DEALER_ID REFERENCES acai_gyoungm(no)

나) acai_receipt_main

- PRIMARY KEY: REC_NUM
- FOREIGN KEY: ID REFERENCES acai_user (ID)
- FOREIGN KEY : bank_code REFERENCES acai_nonghup (CODE)
- FOREIGN KEY: location_code REFERENCES acai_zipcode (NO)
- FOREIGN KEY: REC_NUM REFERENCES acai_rec_num (REC_NUM)

다) acai_receipt_item

- FOREIGN KEY: ITEM_CODE REFERENCES acai_item (CODE)

라) acai_receipt_item_changed

- FOREIGN KEY: ITEM_CODE REFERENCES acai_item (CODE)
- FOREIGN KEY: CHARGE REFERENCES acai_user (ID)

- FOREIGN KEY: GRADE_CODE REFERENCES acai_quality (code)
- FOREIGN KEY: REC_NUM REFERENCES acai_rec_num (REC_NUM)

마) acai_rec_num

- PRIMARY KEY: REC_NUM
- FOREIGN KEY: ID REFERENCES acai_user (ID)

바) acai_user

- PRIMARY KEY: ID
- FOREIGN KEY: ADDRESS REFERENCES acai_zipcode (NO)

사) acai_nong_ex

- PRIMARY KEY: ID
- FOREIGN KEY: ORIGIN REFERENCES acai_zipcode(NO)
- FOREIGN KEY: ID REFERENCES acai_user(ID)
- FOREIGN KEY: NONGHUP REFERENCES acai_nonghup(CODE)
- FOREIGN KEY: BANK REFERENCES acai_bank(code)

아) acai_manage_num

- PRIMARY KEY: rec_num
- FOREIGN KEY: rec_num REFERENCES acai_rec_num(REC_NUM)

자) acai_delete

- PRIMARY KEY: rec_num
- FOREIGN KEY: rec_num REFERENCES acai_rec_num(REC_NUM)
- FOREIGN KEY: admin_ID REFERENCES acai_user(ID)

차) acai_bank

- PRIMARY KEY: code

카) acai_grade

- PRIMARY KEY: CODE

타) acai_gyoungm

- PRIMARY KEY: no

파) acai_item

- PRIMARY KEY: CODE
- FOREIGN KEY: SU_CODE REFERENCES acai_pummok(CODE)

하) acai_item_main

- PRIMARY KEY: CODE

가 -2) acai_pummok

- PRIMARY KEY: CODE

나 -2) acai_load_dc

- PRIMARY KEY: grade

마. Table 구축

물리적 테이블을 구성하여 각각의 코드값을 입력하였다. 코드는 농림부에서 사용하는 코드를 기반으로 하였다. 각 테이블의 구성은 다음과 같다. 테이블의 Naming Rule은 Schema+'_'+의미있는 이름으로 구성하였다. 다음은 본 연구에서 사용되어진 테이블에 관한 정의를 나타낸다.

1) acai_receipt_auction: 수탁판매 접수증의 접수 및 관리에 관련된 테이블이다. Script는 다음과 같다. 'PRIMARY KEY'로 'Rec_num' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) REC_NUM: 접수 번호를 나타낸다. int 형이다.

나) CARGO_FEE: 하역비를 나타낸다. int 형이다.

다) EXPENSE: 비용을 나타낸다. int 형이다.

라) MANAGE_NUM: 관리번호를 나타낸다. varchar 형이다.

마) TRUCK_ID: 트럭 ID를 나타낸다. int 형이다.

바) ACCOUNT_NUM: 은행 계좌번호를 나타낸다. varchar 형이다.

사) DEALER_ID: 경매사 ID를 나타낸다. int 형이다.

아) MANAGER_ID: 관리자 ID를 나타낸다. int 형이다.

자) LOW_COST: 최저 금액을 나타낸다. int 형이다.

차) REST: 기타를 나타낸다. varchar 형이다.

카) RECEIPT: 접수를 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_receipt_auction` (
`REC_NUM` int(11) NOT NULL default '0',
`CARGO_FEE` int(11) NOT NULL default '0',
`EXPENSE` int(11) NOT NULL default '0',
`MANAGE_NUM` varchar(14) default '0',
`TRUCK_ID` int(11) default '0',
`ACCOUNT_NUM` varchar(30) default ' ',
`DEALER_ID` int(11) default '0',
`MANAGER_ID` int(11) default '0',
`LOW_COST` int(11) default '0',
`REST` varchar(10) default NULL,
`RECEIPT` varchar(8) default NULL,
PRIMARY KEY  (`REC_NUM`),
KEY `FK_acai_receipt_auction_2` (`TRUCK_ID`),
KEY `FK_acai_receipt_auction_3` (`DEALER_ID`),
KEY `MANAGER_ID` (`MANAGER_ID`),
CONSTRAINT `acai_receipt_auction_ibfk_1` FOREIGN KEY (`MANAGER
_ID`) REFERENCES `acai_manager` (`no`),
CONSTRAINT `FK_acai_receipt_auction_1` FOREIGN KEY (`REC_NUM`)
```

```
REFERENCES `acai_rec_num` (`REC_NUM`),
CONSTRAINT `FK_acai_receipt_auction_2` FOREIGN KEY (`TRUCK_I
D`) REFERENCES `acai_truck` (`no`),
CONSTRAINT `FK_acai_receipt_auction_3` FOREIGN KEY (`DEALER_I
D`) REFERENCES `acai_gyoungm` (`no`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

2) acai_receipt_main: 출하자의 개인 정보와 관련된 수탁판매 접수증 내용이다. 'PRIMARY KEY'로 'Rec_num' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) REC_NUM: 접수 번호를 나타낸다. int 형이다.

나) ID: 출하자 ID를 나타낸다. varchar 형이다.

다) ship_method: 하역방법을 나타낸다. int 형이다.

라) bank_code: 은행코드를 나타낸다. varchar 형이다.

마) jakmokban: 작목반을 나타낸다. varchar 형이다.

바) location_code: 주소코드를 나타낸다. int 형이다.

사) QCNum: 인증번호를 나타낸다. varchar 형이다.

아) ship_date: 상장 일을 나타낸다. varchar 형이다.

자) ship_time: 상장 시간을 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_receipt_main` (
`REC_NUM` int(11) NOT NULL default '0',
`ID` varchar(8) default NULL,
`ship_method` int(11) NOT NULL,
`bank_code` varchar(6) default '0',
`jakmokban` varchar(10) default NULL,
`location_code` int(11) default NULL,
`QCNum` varchar(8) default NULL,
`ship_date` varchar(8) default NULL,
```

```
`ship_time` varchar(8) default NULL,
PRIMARY KEY (`REC_NUM`),
KEY `ID` (`ID`),
KEY `bank_code` (`bank_code`),
KEY `location_code` (`location_code`),
CONSTRAINT `acai_receipt_main_ibfk_1` FOREIGN KEY (`ID`) REFERE
NCES `acai_user` (`ID`),
CONSTRAINT `acai_receipt_main_ibfk_2` FOREIGN KEY (`bank_code`) R
EFERENCES `acai_nonghup` (`CODE`),
CONSTRAINT `acai_receipt_main_ibfk_3` FOREIGN KEY (`location_cod
e`) REFERENCES `acai_zipcode` (`NO`),
CONSTRAINT `FK_acai_receipt_main_1` FOREIGN KEY (`REC_NUM`)
REFERENCES `acai_rec_num` (`REC_NUM`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

3) acai_receipt_item: 출하되는 농산물을 저장하는 부분이다. 컬럼에 대한 설명은 다음과 같다.

가) REC_NUM: 접수번호를 나타낸다. int 형이다.

나) SEQ: 순번을 나타낸다. int 형이다.

다) ITEM_CODE: 농산물 코드를 나타낸다. varchar 형이다.

라) GRADE_CODE: 등급코드를 나타낸다. varchar 형이다.

마) WEIGHT: 무게 코드를 나타낸다. int 형이다.

바) QUANTITY: 수량을 나타낸다. int 형이다.

사) GWASU: 과수를 나타낸다. int 형이다.

```
CREATE TABLE `acai_receipt_item` (
`REC_NUM` int(11) default NULL,
`SEQ` int(11) NOT NULL default '0',
`ITEM_CODE` varchar(10) default NULL,
`GRADE_CODE` varchar(5) default NULL,
```

```
`WEIGHT` int(11) NOT NULL default '0',
`QUANTITY` int(11) NOT NULL default '0',
`GWASU` int(11) NOT NULL default '0',
KEY `FK_acai_receipt_item_recNum` (`REC_NUM`),
KEY `FK_acai_receipt_item_2` (`ITEM_CODE`),
CONSTRAINT `FK_acai_receipt_item_1` FOREIGN KEY (`REC_NUM`) R
EFERENCES `acai_rec_num` (`REC_NUM`),
CONSTRAINT `FK_acai_receipt_item_2` FOREIGN KEY (`ITEM_CODE`)
REFERENCES `acai_item` (`CODE`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

4) acai_receipt_item_changed: 접수자에 의해 물품이 변경되어졌을 경우의 변경 물품의 정보를 저장한다. 이 테이블을 통하여 접수증의 내용이 변경되었을 경우 그 변경을 사항을 조회할 수 있으며 이를 통하여 추후에라도 발생을 분쟁을 차단할 수 있다. 컬럼에 대한 설명은 다음과 같다.

가) REC_NUM: 접수 번호를 나타낸다. int 형이다.

나) SEQ: 순번을 나타낸다. int 형이다.

다) ITEM_CODE: 농산물 코드를 나타낸다. varchar 형이다.

라) GRADE_CODE: 등급 코드를 나타낸다. varchar 형이다.

마) WEIGHT: 무게 코드를 나타낸다. int 형이다.

바) QUANTITY: 수량을 나타낸다. int 형이다.

사) GWASU: 과수를 나타낸다. int 형이다.

아) CHARGE: 변경자 ID를 나타낸다. varchar 형이다.

자) change_yyyymmdd: 변경일을 나타낸다. varchar 형이다.

차) change_hhmmdd: 변경 시간을 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_receipt_item_changed` (
`REC_NUM` int(11) default NULL,
`SEQ` int(11) NOT NULL,
```

```
`ITEM_CODE` varchar(6) default NULL,
`GRADE_CODE` varchar(5) default NULL,
`WEIGHT` int(11) NOT NULL,
`QUANTITY` int(11) NOT NULL,
`GWASU` int(11) NOT NULL,
`CHARGE` varchar(8) NOT NULL,
`change_yyyymmdd` varchar(8) NOT NULL,
`change_hhmmdd` varchar(6) NOT NULL,
KEY `FK_acai_receipt_item_changed_1` (`REC_NUM`),
KEY `ITEM_CODE` (`ITEM_CODE`),
KEY `CHARGE` (`CHARGE`),
KEY `GRADE_CODE` (`GRADE_CODE`),
CONSTRAINT `acai_receipt_item_changed_ibfk_1` FOREIGN KEY (`ITE
M_CODE`) REFERENCES `acai_item` (`CODE`),
CONSTRAINT `acai_receipt_item_changed_ibfk_2` FOREIGN KEY (`CHA
RGE`) REFERENCES `acai_user` (`ID`),
CONSTRAINT `acai_receipt_item_changed_ibfk_3` FOREIGN KEY (`GRA
DE_CODE`) REFERENCES `acai_quality` (`code`),
CONSTRAINT `FK_acai_receipt_item_changed_1` FOREIGN KEY (`REC_
NUM`) REFERENCES `acai_rec_num` (`REC_NUM`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

5) acai_rec_num: 접수 번호를 저장하는 테이블로 실제 접수와 관련 없이 접수요청이 있을 경우 무조건 증가한다. 'PRIMARY KEY'로 'Rec_num' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) REC_NUM: 접수 번호를 나타낸다. int 형이다.

나) ID: 출하자 ID를 나타낸다. varchar 형이다.

다) CREATE_DT: 접수증 작성 일을 나타낸다. datetime 형이다.

라) COMPLETE: 입하 확인 여부를 나타낸다. tinyint 형이다.

마) COMPLETE_DATE: 입하 확인 시 날짜를 나타낸다. datetime 형이다.

바) COMPLETE_ID: 입하 확인을 한 접수자 ID를 나타낸다. varchar 형이다.

사) status: 상태를 나타낸다. int 형이다.

아) changed: 변경 여부를 나타낸다. tinyint 형이다.

```
CREATE TABLE `acai_rec_num` (
`REC_NUM` int(11) NOT NULL default '0',
`ID` varchar(8) default NULL,
`CREATE_DT` datetime default NULL,
`COMPLETE` tinyint(4) default NULL,
`COMPLETE_DATE` datetime default NULL,
`COMPLETE_ID` varchar(8) default NULL,
`status` int(1) default '0',
`changed` tinyint(1) default '0',
PRIMARY KEY  (`REC_NUM`),
KEY `FK_acai_rec_num_1` (`ID`),
CONSTRAINT `FK_acai_rec_num_1` FOREIGN KEY (`ID`) REFERENCE
S `acai_user` (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

6) acai_manage_num: 관리 번호를 저장하는 곳으로 접수 번호를 Foreign Key로 가지며 실제 등록이 이루어질 때만 증가한다. 'PRIMARY KEY'로 'Rec_num' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) yyyy: 연도를 나타낸다. varchar 형이다.

나) mm: 월을 나타낸다. varchar 형이다.

다) num: 일련번호를 나타낸다. varchar 형이다.

라) reg_date: 등록일을 나타낸다. date 형이다.

마) reg_user: 출하자 ID를 나타낸다. varchar 형이다.

바) rec_num: 접수 번호를 나타낸다. int 형이다.

```
CREATE TABLE `acai_manage_num` (
`yyyy` varchar(4) default NULL,
`mm` varchar(2) default NULL,
`num` varchar(6) default NULL,
`reg_date` date default NULL,
`reg_user` varchar(8) default NULL,
`rec_num` int(11) NOT NULL,
PRIMARY KEY (`rec_num`),
CONSTRAINT `FK_acai_manage_num_1` FOREIGN KEY (`rec_num`) RE
FERENCES `acai_rec_num` (`REC_NUM`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

7) acai_user: 사용자 정보를 저장하는 테이블이다. 'GRADE' 컬럼의 경우 출하자는 '0', 접수자는 '1', 관리자는 '2'이다. 'PRIMARY KEY'로 'ID' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) ID: 사용자 ID를 나타낸다. varchar 형이다.

나) NAME: 사용자 이름을 나타낸다. varchar 형이다.

다) REG_NUM: 접수 번호를 나타낸다. varchar 형이다.

라) ADDRESS: 사용자 주소 코드를 나타낸다. int 형이다.

마) PHONE: 사용자 전화번호를 나타낸다. varchar 형이다.

바) MOBILE: 사용자 휴대 전화번호를 나타낸다. varchar 형이다.

사) REG_DATE: 등록일을 나타낸다. date 형이다.

아) GRADE: 등급코드를 나타낸다. int 형이다.

자) PASSWORD: 비밀번호를 나타낸다. varchar 형이다.

차) Email: 이메일 주소를 나타낸다. varchar 형이다.

카) Confirm: 관리자 확인 여부를 나타낸다. tinyint 형이다.

타) ADDRESS2: 주소 나머지 부분을 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_user` (
`ID` varchar(8) NOT NULL default '',
`NAME` varchar(20) default NULL,
`REG_NUM` varchar(14) default NULL,
`ADDRESS` int(5) default NULL,
`PHONE` varchar(15) default NULL,
`MOBILE` varchar(15) default NULL,
`REG_DATE` date default NULL,
`GRADE` int(11) default NULL,
`PASSWORD` varchar(8) default NULL,
`Email` varchar(40) default NULL,
`Confirm` tinyint(4) default NULL,
`ADDRESS2` varchar(50) default NULL,
PRIMARY KEY  (`ID`),
KEY `ADDRESS` (`ADDRESS`),
CONSTRAINT `acai_user_ibfk_1` FOREIGN KEY (`ADDRESS`) REFERE
NCES `acai_zipcode` (`NO`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

8) acai_delete: 삭제되어진 접수에 관한정보를 저장하는 테이블로 접수번호를 Foreign Key로 갖는다. 'PRIMARY KEY'로 'Rec_num' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) rec_num: 접수 번호를 나타낸다. int 형이다.

나) admin_ID: 관리자 ID를 나타낸다. varchar 형이다.

다) delete_YYYYMMDD: 삭제 연 월 일을 나타낸다. varchar 형이다.

라) delete_HHMMDD: 삭제 시 분 초를 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_delete` (
`rec_num` int(11) NOT NULL,
```

```
`admin_ID` varchar(8) NOT NULL,
`delete_YYYYMMDD` varchar(8) default NULL,
`delete_HHMMDD` varchar(6) default NULL,
PRIMARY KEY (`rec_num`),
KEY `FK_acai_delete_2` (`admin_ID`),
CONSTRAINT `FK_acai_delete_1` FOREIGN KEY (`rec_num`) REFEREN
CES `acai_rec_num` (`REC_NUM`),
CONSTRAINT `FK_acai_delete_2` FOREIGN KEY (`admin_ID`) REFERE
NCES `acai_user` (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

9) acai_bank: 은행 코드를 관리하는 테이블이다. 'PRIMARY KEY'로 'CODE' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) code: 코드를 나타낸다. varchar 형이다.

나) name: 코드 이름을 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_bank` (
`code` varchar(2) NOT NULL,
`name` varchar(20) NOT NULL,
PRIMARY KEY (`code`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

10) acai_grade: 농산물의 등급을 관리하는 테이블이다. 'PRIMARY KEY'로 'CODE' 컬럼을 사용한다.

가) code: 코드를 나타낸다. varchar 형이다.

나) name: 코드 이름을 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_grade` (
```

```
'CODE' varchar(4) NOT NULL default '',
'NAME' varchar(20) default NULL,
PRIMARY KEY  ('CODE')
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

11) acai_gyoungm: 경매사를 관리하는 테이블이다. 'PRIMARY KEY'로 'no' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) no: 코드를 나타낸다. int 형이다.

나) name: 경매사 이름을 나타낸다. varchar 형이다.

다) useYN: 사용여부를 나타낸다. tinyint 형이다.

```
CREATE TABLE 'acai_gyoungm' (
'no' int(4) NOT NULL auto_increment,
'name' varchar(20) NOT NULL,
'useYN' tinyint(1) NOT NULL default '1',
PRIMARY KEY  ('no')
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

12) acai_item: 농산물 코드를 관리하는 테이블이다. 품종에 관한 코드를 관리한다. 품목 코드를 Foreign Key로 갖는다. 'PRIMARY KEY'로 'CODE' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) CODE: 코드를 나타낸다. varchar 형이다.

나) NAME: 품종 이름을 나타낸다. varchar 형이다.

다) SU_CODE: 품종에 상응하는 품목 코드를 나타낸다. varchar 형이다.

```
CREATE TABLE 'acai_item' (
'CODE' varchar(6) NOT NULL default '0',
```

```
'NAME' varchar(20) NOT NULL default '',
'SU_CODE' varchar(6) NOT NULL,
PRIMARY KEY  ('CODE'),
KEY 'FK_acai_item_1' ('SU_CODE'),
CONSTRAINT 'FK_acai_item_1' FOREIGN KEY ('SU_CODE') REFERE
NCES 'acai_pummok' ('CODE')
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

13) acai_item_main: 농산물의 종류를 관리하는 테이블이다. 'PRIMARY KEY'로 'CODE' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) CODE: 코드를 나타낸다. varchar 형이다.

나) NAME: 농산물의 종류를 나타낸다. varchar 형이다.

```
CREATE TABLE 'acai_item_main' (
'CODE' varchar(4) NOT NULL,
'NAME' varchar(10) default NULL,
PRIMARY KEY  ('CODE')
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

14) acai_pummok: 품목 코드를 관리하는 테이블로 종류 코드를 Foreign Key로 갖는다. 'PRIMARY KEY'로 'CODE' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) CODE: 코드를 나타낸다. varchar 형이다.

나) NAME: 품목 이름을 나타낸다. varchar 형이다.

다) SU_CODE: 품목에 상응하는 종류 코드이다. varchar 형이다.

```
CREATE TABLE 'acai_pummok' (
'CODE' varchar(6) NOT NULL,
```

```
'NAME' varchar(20) NOT NULL,
'SU_CODE' varchar(4) NOT NULL,
PRIMARY KEY ('CODE')
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

15) acai_load_dc: 하역비의 DC율을 관리하는 테이블이다. 'PRIMARY KEY'로 'grade' 컬럼을 사용한다.

가) grade: 등급을 나타낸다. tinyint 형이다.

나) discount: 할인율을 나타낸다. int 형이다.

다) weight: 무게를 나타낸다. int 형이다.

```
CREATE TABLE 'acai_load_dc' (
'grade' tinyint(3) NOT NULL default '0',
'discount' int(2) default NULL,
'weight' int(4) default NULL,
PRIMARY KEY ('grade')
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

16) acai_load_fee: 하역비를 관리하는 테이블이다. 컬럼에 대한 설명은 다음과 같다.

가) weight: 무게를 나타낸다. int 형이다.

나) unit_cost: 단위당 비용을 나타낸다. int 형이다.

```
CREATE TABLE 'acai_load_fee' (
'weight' int(3) default NULL,
'unit_cost' int(5) default NULL
```

```
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

17) acai_manager: 책임자를 관리하는 테이블이다. 'PRIMARY KEY'로 'no' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) no: 코드 번호를 나타낸다. int 형이다.

나) name: 책임자 이름을 나타낸다. varchar 형이다.

다) useYN: 사용 여부를 나타낸다. tinyint 형이다.

```
CREATE TABLE `acai_manager` (
`no` int(4) NOT NULL,
`name` varchar(20) default NULL,
`useYN` tinyint(4) default NULL,
PRIMARY KEY (`no`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

18) acai_nong_ex: 출하자의 추가 정보를 관리하는 테이블이다. 'PRIMARY KEY'로 'ID' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) ID: 출하자 ID를 나타낸다. varchar 형이다.

나) ORIGIN: 원산지 코드를 나타낸다. int 형이다.

다) NONGHUP: 농협 코드를 나타낸다. varchar 형이다.

라) QCMNum: 인증번호를 나타낸다. varchar 형이다.

마) JAKMOKBAN: 작목반을 나타낸다. varchar 형이다.

바) BANK: 은행을 나타낸다. varchar 형이다.

사) ACCOUNT: 계좌 번호를 나타낸다. varchar 형이다.

아) ACCOUNT_NAME: 계좌 번호의 소유자 성명을 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_nong_ex` (
```

```
`ID` varchar(8) NOT NULL,
`ORIGIN` int(5) default '0',
`NONGHUP` varchar(6) default '000000',
`QCMNum` varchar(10) default ' ',
`JAKMOKBAN` varchar(20) default ' ',
`BANK` varchar(2) default '00',
`ACCOUNT` varchar(20) default ' ',
`ACCOUNT_NAME` varchar(20) default ' ',
PRIMARY KEY  (`ID`),
KEY `FK_acai_nong_ex_3` (`NONGHUP`),
KEY `FK_acai_nong_ex_4` (`BANK`),
KEY `ORIGIN` (`ORIGIN`),
CONSTRAINT `acai_nong_ex_ibfk_1` FOREIGN KEY (`ORIGIN`) REFER
ENCES `acai_zipcode` (`NO`),
CONSTRAINT `FK_acai_nong_ex_1` FOREIGN KEY (`ID`) REFERENCE
S `acai_user` (`ID`),
CONSTRAINT `FK_acai_nong_ex_3` FOREIGN KEY (`NONGHUP`) REF
ERENCES `acai_nonghup` (`CODE`),
CONSTRAINT `FK_acai_nong_ex_4` FOREIGN KEY (`BANK`) REFERE
NCES `acai_bank` (`code`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

19) acai_nonghup: 농협 코드를 관리하는 테이블이다. 'PRIMARY KEY'로 'COD E' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) CODE: 코드를 나타낸다. varchar 형이다.

나) ADDRESS: 농협 주소를 나타낸다. varchar 형이다.

다) NAME: 농협 이름을 나타낸다. varchar 형이다.

```
CREATE TABLE `acai_nonghup` (
```

```
'CODE' varchar(6) NOT NULL default '0',
'ADDRESS' varchar(40) NOT NULL default '',
'NAME' varchar(20) NOT NULL default '',
PRIMARY KEY ('CODE')
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

20) acai_quality: 품질 코드를 관리하는 테이블이다. 'PRIMARY KEY'로 'CODE' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) code: 코드를 나타낸다. varchar 형이다.

나) name: 품질을 나타낸다. varchar 형이다.

```
CREATE TABLE 'acai_quality' (
'code' varchar(4) NOT NULL default '',
'name' varchar(20) default NULL,
PRIMARY KEY ('code')
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

21) acai_truck: 하역에 이용된 트럭코드를 관리하는 테이블이다. 'PRIMARY KEY'로 'no' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) no: 코드를 나타낸다. int 형이다.

나) num: 트럭 번호를 나타낸다. varchar 형이다.

다) useYN: 사용 여부를 나타낸다. tinyint 형이다.

```
CREATE TABLE 'acai_truck' (
'no' int(11) NOT NULL auto_increment,
'num' varchar(20) default NULL,
'useYN' tinyint(4) default NULL,
PRIMARY KEY ('no')
```

```
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

22) acai_weight: 중량 코드를 관리하는 테이블이다. 'PRIMARY KEY'로 'no' 컬럼을 사용한다. 컬럼에 대한 설명은 다음과 같다.

가) no: 코드를 나타낸다. int 형이다.

나) NAME: 중량을 나타낸다. varchar 형이다.

다) useYN: 사용여부를 나타낸다. tinyint 형이다.

```
CREATE TABLE `acai_weight` (
`no` int(11) NOT NULL,
`NAME` varchar(10) default NULL,
`useYN` tinyint(4) default NULL,
PRIMARY KEY (`no`)
) ENGINE=InnoDB DEFAULT CHARSET=euckr
```

바. Stored Procedure

연속된 Query의 실행이 필요하거나 Query의 결과를 Query의 변수로 사용하는 부분에서는 성능 향상과 시스템의 안정적 운영을 위해 Stored Procedure를 이용하였다. 이를 통하여 안정적인 Logic을 구현 하였다.

1) deleteReceipt

입력으로 접수 번호 사용자 ID를 받으며 결과로는 acai_rec_num 테이블의 해당 접수번호의 상태 값을 변경하고 acai_delete에 해당 항목을 추가한다.

```
CREATE PROCEDURE `deleteReceipt`(recNum Integer, ID varchar(8))
BEGIN
update acai_rec_num
set status=2
```

```
where rec_num=recNum;
insert into acai_delete
values(recNum,ID,DATE_FORMAT(curdate(),'%Y%m%d'),DATE_FORMA
T(curdate(),'%h%i'));
END
```

2) deleteServRecNum

사용자에 의한 등록정보 삭제를 시행한다. 접수번호를 입력으로 받으며 출력은 등록정보를 기준으로 항목을 삭제한다.

```
CREATE PROCEDURE `deleteServRecNum`(IN recNum INTEGER)
BEGIN
delete from acai_receipt_item where rec_num=recNum;
delete from acai_receipt_auction where rec_num=recNum;
END
```

3) getLoadFee

하역비를 계산한다. 하역비는 출하자가 항목을 입력하면 자동으로 계산되어지도록 구성하였다. 입력은 항목 코드와 수량을 기준으로 하며 acai_load_dc 테이블에서 중량을 기준으로 DC율을 가져오고 acai_load_fee에서 항목에 맞는 비용을 가져와서 계산한다.

```
CREATE FUNCTION `getLoadFee`(code int, boxNum int) RETURNS int
BEGIN
DECLARE  fee int;
DECLARE  dc int ;
DECLARE  uCost int ;
```

```
select max(discount) into dc
from acai_load_dc
where weight<= ( select name*boxNum
                 from acai_weight
                 where no=code);
select unit_cost into uCost
from acai_load_fee
where weight= code;
select uCost*boxNum*(1-dc/100) into fee;
return fee;
END
```

4) getManageNum

출하자가 등록을 하면 접수번호를 기준으로 관리번호를 생성하여 저장하고 그 값을 반환한다. 반환값은 년 월+일련번호로 구성되어지며 실제 등록되어진 건수 만큼 일련번호가 증가되어지도록 하였다. 따라서 접수번호는 실제 등록 건수와 무관하게 등록시도만 있어도 증가되고 관리번호는 실제 등록이 이루어 져야 증가한다.

```
CREATE FUNCTION `getManageNum`(recNum int) RETURNS char(14)
BEGIN
DECLARE manageNum CHAR(14);
DECLARE x VARCHAR(10);
select id into x
from acai_rec_num
where rec_num=recNum;
SELECT max(num)+1 INTO manageNum
FROM acai_manage_num ;
insert into acai_manage_num
values(DATE_FORMAT(curdate(),'%Y'),DATE_FORMAT(curdate(),'%m'),
```

```
LPAD(manageNum,6,'0'),curdate(),x,recNum);
select  CONCAT(DATE_FORMAT(curdate(),'%Y'),'-',DATE_FORMAT(cu
rdate(),'%m'),'-', LPAD(manageNum,6,'0') ) into manageNum;
return manageNum;
END
```

5) getRecNum

접수번호를 가져온다. 접수번호는 사용자 ID를 입력받아 증가 하도록 되어있다.

```
CREATE FUNCTION `getRecNum`(x VARCHAR(10)) RETURNS int
BEGIN
DECLARE  recNum int ;
SELECT max(rec_num)+1 INTO recNum
FROM acai_rec_num ;
insert into acai_rec_num
values(recNum,x,concat(curdate(),' ',CURTIME()),0,curdate(),'',0,0);
return recNum;
END
```

6) getStatus

해당 접수번호에 해당하는 상태 값을 반환한다. 입력으로 접수 번호를 받는다.

```
CREATE FUNCTION `getStatus`(recNum int) RETURNS int
begin
DECLARE  st int;
select status into st
from acai_rec_num
where rec_num=recNum;
```

```
return st;
end
```

사. SQL(Structured Query Language)

위에서 정의되어진 업무 분석을 통해 본 연구에서 사용되어질 query문을 정리하여 먼저 시스템의 적용 여부를 결정 하였다.

1) 사용자 관리

사용자 관련 쿼리는 사용자 등록 삭제 수정과 관련된 쿼리문이다.

가) 사용자 찾기

입력 값으로 사용자 ID를 사용하며 이를 통하여 사용자 정보를 반환한다.

```
SELECT a.NAME,a.REG_NUM, concat(b.sido,' ',b.gugun,' ',b.dong) addres
s, a.address2, a.address addcode,a.phone, a.MOBILE, a.PASSWORD,a.email,
a.grade,a.confirm
FROM acai_USER a, acai_zipcode b
WHERE a.address=b.no
and a.id=?
```

나) 사용자 추가정보 찾기

입력 값으로 사용자 ID를 이용하며 사용자의 추가 정보를 반환한다.

```
SELECT a.id,a.origin,concat(c.sido,' ',c.gugun,' ',c.dong) address ,a.nonghu
p nonghup,b.name no_Name,a.QCMNum,a.jakmokban,a.bank,d.name ba_nam
e,a.account,a.account_name
FROM  acai_nong_ex a,acai_nonghup b,acai_zipcode c, acai_bank d
```

```
WHERE   a.origin=c.no
and     a.nonghup=b.code
and     a.bank=d.code
and     a.id=?
```

다) 사용자 리스트를 반환한다.

```
SELECT   id, name,phone,mobile, email,confirm,grade
FROM acai_USER
where confirm=true
```

라) Login

사용자 Login을 확인한다. 입력 값으로 사용자 ID와 비밀번호를 받으며 반환 값으로 사용자 등급을 사용한다.

```
select grade
from acai_user
where id=?
and password=?
```

마) 사용자 존재 여부 판단

인자로 전달되는 아이디를 가지는 사용자가 존재하는지의 유무를 판별한다. 반환 값으로 존재할 경우 1을 존재하지 않을 경우 0을 반환한다.

```
SELECT count(*) FROM acai_USER
WHERE id=?
```

2) 코드 관리

가) 주소 찾기

입력 값으로 받아들인 주소의 일부 문자를 통해 이에 해당하는 주소들을 반환한다.

```
SELECT no,zipcode,sido,gugun,dong,bunji
FROM acai_zipcode
where dong like concat('%',?,'%')
```

나) 농협 찾기

농협 주소의 일부를 입력으로 받은 후 이에 해당하는 농협들을 반환한다.

```
SELECT code, address, name
 FROM acai_nonghup
 where address like concat('%',?,'%')
```

다) 코드를 통한 주소 찾기

입력으로 주소 코드를 입력 받고 이에 맞는 주소 값을 반환한다.

```
SELECT no,zipcode,sido,gugun,dong,bunji
 FROM acai_zipcode
 where no=?
```

라) 품목 찾기

입력 값으로 코드 값을 받고 이에 대한 품목 코드를 반환한다.

```
select su_code
 from acai_item
 where code=?
```

마) 입력 값으로 ID를 받으며 이에 대한 사용자 정보를 반환한다.

```
select *
from acai_user
where id=?
```

바) 접수증 검색

접수 번호를 입력으로 받으며 이에 대한 접수증 정보를 반환한다.

```
select *
from acai_receipt_main
where rec_num=?
```

사) 품종 검색

코드를 입력 값으로 받으며 코드에 맞는 품종 이름을 반환한다.

```
 select name
 from acai_item
 where code=?
```

아) 등급 검색

코드 값을 입력으로 받으며 이에 대한 등급 이름을 반환한다.

```
select name
from acai_quality
where code=?
```

자) 경매사

등록되어진 경매사에 대한 정보를 반환한다.

```
select *
from acai_gyoungm
where useYN=true
```

차) 관리자

등록되어진 접수증상의 관리자에 대한 정보를 반환한다.

```
select *
from acai_manager
where useYN=true
```

카) 트럭 정보

접수되어진 트럭에 대한 정보를 반환한다.

```
select *
from acai_truck
```

```
where useYN=true
```

타) 무게 정보

무게 정보를 반환한다.

```
select *
from acai_weight
where useYN=true
```

파) 코드 조회

입력으로 테이블 값을 받으며 이에 대한 정보를 반환한다. 일반 코드 관리에 사용되어진다.

```
select *
from "+tableName);
where no>0
```

하) 코드 수정

입력으로 테이블 값을 받으며 이에 대한 정보를 수정한다.

```
update "+tableName);
set useYN=?
where no=?
```

가-1) 코드 입력

신규 코드값을 입력한다.

```
INSERT INTO "+tableName
 values(?, ?, ?)
```

3) 접수증 관리

접수증 관리와 관련된 SQL을 나타낸다.

가) 입력으로 접수 번호를 받으며 해당 접수 번호에 대한 데이터를 삭제한다.

```
delete from "+table+" where rec_num=?
```

나) 변경 여부 수정

입력으로 접수 번호를 받으며 해당 접수증의 내용이 변경 여부를 수정한다.

```
update acai_rec_num
set changed=true
where rec_num=?
```

다) 변경 log 저장

접수자에 의한 변경 여부를 저장한다. 입력 값으로 변경한 접수자 ID와 접수증 ID를 받는다.

```
insert into acai_receipt_item_changed
select REC_NUM,SEQ,ITEM_CODE,GRADE_CODE,WEIGHT, QUANTITY,
GWASU,?,DATE_FORMAT(curdate(),'%Y%m%d'),CURTIME()+0
```

```
from acai_receipt_item
where rec_num=?)
```

라) 접수증의 개인정보 저장

접수증의 개인정보를 저장한다. 입력 값으로 접수증의 개인정보 내용을 받는다.

```
INSERT INTO ACAI_RECEIPT_MAIN
values (?, ?, ?, ?, ?, ?,?,?,?)
```

마) 접수증의 기타정보 저장

접수증의 기타정보를 저장한다. 입력 값으로 접수증의 기타정보 내용을 받는다.

```
INSERT INTO ACAI_RECEIPT_AUCTION(REC_NUM,CARGO_FEE,EXP
ENSE,MANAGE_NUM, TRUCK_ID,ACCOUNT_NUM,DEALER_ID,MANA
GER_ID,LOW_COST,REST,RECEIPT
values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
```

바) 접수증의 농산물 정보 저장

접수증의 농산물 정보를 저장한다. 입력값으로 접수증의 농산물 정보 내용을 받는다.

```
INSERT INTO ACAI_RECEIPT_item values
(?, ?, ?, ?, ?, ?, ?)
for(i=1;i<rProducts.length;i++)
{
```

```
        insertQuery.append(",(?, ?, ?, ?, ?, ?, ?)
    }
```

사) 농산물 정보 조회

농산물 정보를 조회한다. 입력 값으로 품종 code 값을 받는다.

```
SELECT code,name
FROM acai_item
where substring(su_code,1,2)=?
```

아) 농산물 등급

농산물 등급을 반환한다.

```
SELECT code,name
FROM acai_quality
order by code
```

자) 현재 접수증 현황

현재 등록된 접수증 정보 반환

```
SELECT code,name
from acai_item_main
order by name
```

차) 신규 접수 요청

신규 접수 요청이 있을 경우 getRecNum 프로시져를 호출하여 새로운 접수 번호를 반환 받는다. 접수 번호는 요청이 있으며 무조건 생성된다.

```
SELECT ACAI.getRecNum(?) recNum
```

카) 관리 번호 생성

입력 값으로 접수 번호를 받으며 관리 번호를 반환한다. 출하자가 접수를 했을 경우에만 생성되며 키 값으로는 접수 번호를 사용한다.

```
SELECT ACAI.getManageNum(?) manageNum
```

타) 하역비 생성

하역비를 계산하며 입력 값으로 수량과 무게를 받는다. 반환 값은 하역비 이다.

```
SELECT ACAI.getLoadFee(?,?) loadFee
```

파) 농산물 조회

입력으로 출하자 ID를 받으며 해당 출하자에 대한 전체 등록 농산물 리스트를 반환한다.

```
select a.rec_num,d.name item,c.weight,e.name grade,c.quantity,b.ship_date s
hipDate,a.status status,a.changed  change1
 from acai_rec_num a, acai_receipt_main b, acai_receipt_item c,
 acai_item d, acai_quality e
 where a.rec_num=b.rec_num
 and a.rec_num=c.rec_num
 and c.item_code=d.code
 and c.grade_code=e.code
```

```
and a.id=?
order by rec_num
```

하) 농산물 조회 2

출하자가 등록한 농산물 리스트를 조회 한다. 입력 값으로 출하자 ID와 접수증의 상태 값을 받으며 이에 해당하는 농산물 리스트를 반환한다.

```
select a.rec_num,d.name item,c.weight,e.name grade,c.quantity,b.ship_date s
hipDate,a.status status , a.changed change1
 from acai_rec_num a, acai_receipt_main b, acai_receipt_item c,
 acai_item d, acai_quality e
 where a.rec_num=b.rec_num
 and a.rec_num=c.rec_num
 and c.item_code=d.code
 and c.grade_code=e.code
 and a.id=?
 and status=?
 order by a.rec_num
```

가-1) 농산물 조회

출하자가 등록한 농산물 리스트를 조회한다. 입력으로 출하자 ID와 기간을 받으며 이에 해당하는 농산물 리스트를 반환한다.

```
 select a.rec_num,d.name item,c.weight,e.name grade,c.quantity,b.ship_date
shipDate,a.status status, a.changed change1
 from acai_rec_num a, acai_receipt_main b, acai_receipt_item c,
 acai_item d, acai_quality e
 where a.rec_num=b.rec_num
```

```
and a.rec_num=c.rec_num
and c.item_code=d.code
and c.grade_code=e.code
and a.id=?
and ship_date between ? and ?
order by a.rec_num
```

나-1) 접수증 조회

접수증의 농산물을 조회한다. 입력으로 접수 번호를 받으며 이에 해당하는 농산물 리스트를 반환한다.

```
select REC_NUM,SEQ,ITEM_CODE,GRADE_CODE,WEIGHT,QUANTITY,
GWASU
from acai_receipt_item
where rec_num=?
order by seq
```

다-1) 기타 정보 조회

접수 번호를 입력으로 받으며 이에 해당하는 기타 정보를 반환하다.

```
select REC_NUM,CARGO_FEE,EXPENSE,MANAGE_NUM,TRUCK_ID,e.n
um tnum,ACCOUNT_NUM,
DEALER_ID,d.name dname,MANAGER_ID,c.name mname,LOW_COST,RE
ST,receipt, b.name rname
from acai_receipt_auction a,acai_user b, acai_manager c, acai_gyoungm d,
acai_truck e
where a.manager_id=c.no
and a.dealer_id=d.no
```

```
and a.truck_id=e.no
and b.id=a.receipt
and rec_num=?
```

라-1) 접수자 정보 조회

입력으로 접수 번호를 받으며 이에 해당하는 접수자 정보를 반환한다.

```
select a.REC_NUM,a.ID,a.ship_method,a.bank_code,b.name nm,a.jakmokban,
a.location_code,a.QCNum,ship_date,c.status status, c.changed change1
from acai_receipt_main a,acai_nonghup b, acai_rec_num c
where a.rec_num=?
and a.bank_code=b.code
and a.rec_num=c.rec_num
```

마-1) 접수 농산물 조회

입력으로 출하자 ID를 받으며 해당 출하자가 등록한 상품 리스트를 반환한다.

```
select a.rec_num num,d.name nonghup,a.id id, b.name name,b.mobile mobi
le,c.ship_date sd,c.ship_time st
from acai_rec_num a, acai_user b,acai_receipt_main c, acai_nonghup d
where a.rec_num=c.rec_num
and a.id=b.id
and c.bank_code=d.code
and a.status=0
and c.id=?
```

바-1) 입하 확인

접수자에 의해 입하 확인 한다. 입력으로 접수 번호와 접수자 ID를 받는다.

```
update acai_rec_num
set status=1, complete_date=now(), complete_ID=?
where rec_num=?
```

사-1) 모바일 접수자 정보 생성

모바일을 이용한 접수시 접수자 정보를 생성한다. 입력으로 접수 번호화 출하자 ID를 받는다.

```
insert into acai_receipt_main
select ?, a.ID,0,a.nonghup,a.jakmokban,a.origin,a.qcmNum,curdate()+0,curtime()+0
from acai_nong_ex a, acai_user b
where a.id=b.id
and a.id=?
```

아-1) 하역비 조회

하역비를 조회한다. 입력으로 접수 번호를 받는다.

```
select cargo_fee
from acai_receipt_auction
where rec_num=?
```

자-1) 기본 통계

기본 통계 자료를 보여 준다.

```
select count(b.item_code) a,c.name b,d.name c,sum(b.quantity) d,count(e.sta
```

```
tus) e,e.status f
from acai_receipt_main a, acai_receipt_item b, acai_pummok c, acai_item d,
acai_rec_num e
where a.rec_num=b.rec_num
and b.item_code=d.code
and d.su_code=c.code
and e.rec_num=a.rec_num
group by b.item_code,e.status
order by f,b,c
```

차-1) 날짜 기준 기본 통계 자료

날짜를 기준으로 기본 통계 자료를 보여준다. 입력으로 연 월 일 값을 받는다.

```
select count(b.item_code) a,c.name b,d.name c,sum(b.quantity) d,count(e.sta
tus) e,e.status f
from acai_receipt_main a, acai_receipt_item b, acai_pummok c, acai_item d,
acai_rec_num e
where a.rec_num=b.rec_num
and b.item_code=d.code
and d.su_code=c.code
and e.rec_num=a.rec_num
and a.ship_date=?
group by b.item_code,e.status
order by f,b,c
```

카-1) 농산물 품종 변경 자료 조회

농산물의 품종 변경 자료를 조회 한다. 입력으로 접수 번호와 순번을 받는다.

```
select c.name a,d.name b
from acai_receipt_item_changed a, acai_receipt_item b, acai_item c, acai_ite
m d
where a.rec_num=b.rec_num
and a.seq=b.seq
and a.item_code=c.code
and b.item_code=d.code
and b.rec_num=?
and a.seq=?
```

타-1) 농산물 등급 변경 자료 조회

농산물의 등급 변경 자료를 조회 한다. 입력으로 접수 번호와 순번을 받는다.

```
select c.name a,d.name b
from acai_receipt_item_changed a, acai_receipt_item b, acai_quality c, acai_
quality d
where a.rec_num=b.rec_num
and a.seq=b.seq
and a.grade_code=c.code
and b.grade_code=d.code
and b.rec_num=?
and a.seq=?
```

파-1) 농산물 무게 변경 자료 조회

농산물의 무게 변경 자료를 조회한다. 입력으로 접수 번호와 순번을 받는다.

```
select c.name a,d.name b
from acai_receipt_item_changed a, acai_receipt_item b, acai_weight c, acai_
```

```
weight d
where a.rec_num=b.rec_num
and a.seq=b.seq
and a.weight=c.no
and b.weight=d.no
and b.rec_num=?
and a.seq=?
```

하-1) 농산물 과수 변경 자료 조회

농산물의 과수 변경 자료를 조회한다. 입력으로 접수 번호와 순번을 받는다.

```
select a.gwasu a,b.gwasu b
from acai_receipt_item_changed a, acai_receipt_item b
where a.rec_num=b.rec_num
and a.seq=b.seq
and b.rec_num=?
and a.seq=?
```

가-2) 농산물 수량 변경 자료 조회

농산물의 수량 변경 자료를 조회한다. 입력으로 접수 번호와 순번을 받는다.

```
select a.quantity a, b.quantity b
from acai_receipt_item_changed a, acai_receipt_item b
where a.rec_num=b.rec_num
and a.seq=b.seq
and b.rec_num=?
and a.seq=?
```

나-2) 변경 농산물 조회

입력으로 접수 번호를 받으며 해당 접수증의 변경 농산물 리스트를 보여준다.

```
select  b.seq seq,strcmp(a.item_code,b.item_code) item,strcmp(a.grade_code,
b.grade_code) grade,
strcmp(a.weight,b.weight) weight,strcmp(a.quantity,b.quantity) quan,strcmp
(a.gwasu,b.gwasu) gwasu
from acai_receipt_item_changed a, acai_receipt_item b
where a.rec_num=b.rec_num
and a.seq=b.seq
and b.rec_num=?
```

다-2) 모바일 접수증 조회

모바일 사용자용이며 입력으로 접수 번호를 받으며 이에 해당하는 접수증을 반환한다.

```
select REC_NUM,SEQ,b.name item,a.item_code itemCode,c.name pummok,d.
name grade,
a.grade_code gradeCode,e.name weight,a.weight weightCode,QUANTITY,G
WASU
from  acai_receipt_item a, acai_item b, acai_pummok c,acai_quality d,acai_
weight e
where rec_num=?
and a.item_code=b.code
and b.su_code=c.code
and e.no=a.weight
and a.grade_code=d.code
```

3. 서버 시스템 설계 및 구축

본 연구에서는 전체 시스템을 크게 3부분으로 분류하였다. 데이터를 관장하는 DBMS 부분과 Business Logic 및 사용자와의 통신을 담당하는 서버 부분 그리고 사용자의 시스템 접근에 관련된 사용자 부분으로 구분하였다. 먼저 DBMS 부분은 서버와의 통신을 통해 데이터를 저장하고 요청에 의해 데이터를 제공하는 부분이며 서버 부분은 사용자의 요청에 대해 DBMS와의 통신을 통해 Business Logic을 수행하여 그 결과를 사용자에게 제공해 주며 사용자 부분은 서버와의 접근에 대한 인터페이스를 제공한다.

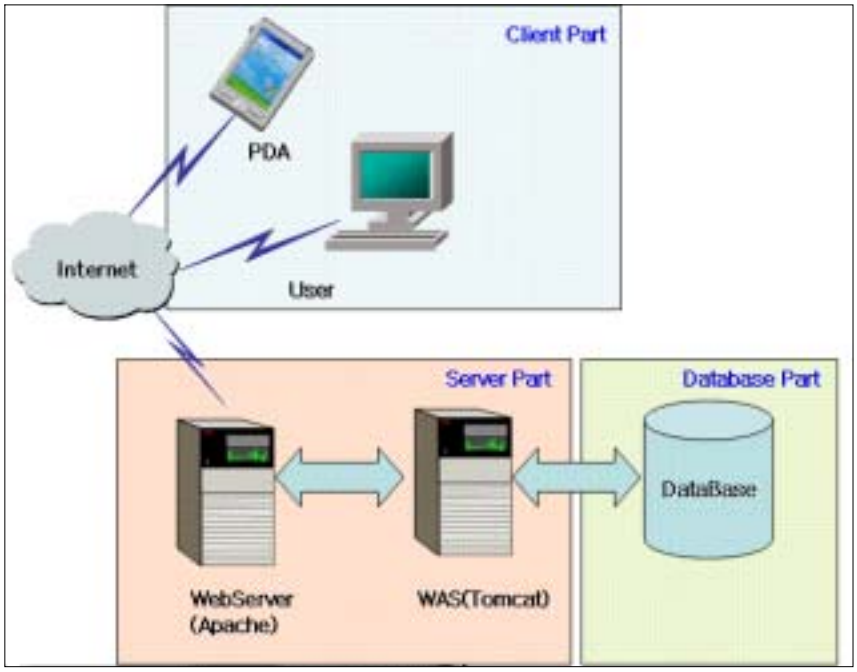

그림 3.3.1 전체 시스템 구성도

가. 전체 시스템 설계

본 연구에서는 클라이언트, 서버, DB로 구성된 3-layer기반으로 시스템을 구축하였으며 클라이언트 부분은 PC상에서 Web Browser를 통한 접근과 PDA상의 WebService를 이용한 접근을 제공하도록 하였다. 또한 WebServer와 WAS 서버 및 DBMS를 분리하여 부하를 분산하도록 유도하였다.

1) 서버 부분

서버 부분은 공개 WAS인 Jakarta-Tomacat을 WAS(Web Application Server)로 사용하였으며 Web Server로는 Apache를 사용하였다.

가) Jakarta-Tomcat

톰캣은 아파치 소프트웨어 재단의 애플리케이션 서버로서, 자바 서블릿을 실행시키고 JSP 코드가 포함되어 있는 웹페이지를 만들어준다. 자바 서블릿과 JSP 규격의 '참조용 구현'으로 평가되고 있는 톰캣은, 개발자들의 개방적 협력 작업의 산물로 바이너리 버전과 소스코드 버전 둘 모두를 아파치 웹사이트에서 얻을 수 있다. 톰캣은 자체적으로 보유하고 있는 내부 웹서버와 함께 독립적으로 사용될 수도 있지만 아파치나 넷스케이프 엔터프라이즈 서버, IIS, 마이크로소프트의 PWS 등 다른 웹서버와 함께 사용될 수도 있다. 톰캣은 흔히 '자카르타 프로젝트'라고 불리는 여러 open source 공동작업 중 하나이다. 본 연구에서는 5.5.9 버전을 사용하였다.

나) Apache

아파치는 1995년 그 당시에 가장 인기 있었던 웹 서버중의 하나인 NCSA HTTPD 1.3 버젼을 기반으로 탄생하였다. 그 후 기존의 NCSA 웹 서버에 더욱 향상된 기능들을 탑재하여 Apache 웹 서버를 발표하였다. 지속적으로 패치파일을 제공하고 최고의 퍼포먼스를 내고 있기 때문이다. 물론 무료로 제공된다는 점과 많은 마켓쉐어의 점유로 인하여 안정성을 인정받았다. 아파치는 현재까지 2.1 까지 발표되었으며, 1.2 시리즈는 테스트 버전이 아닌 안정된 버전이다. 아파치 프로세스 모델은 아래 그림과 같다.

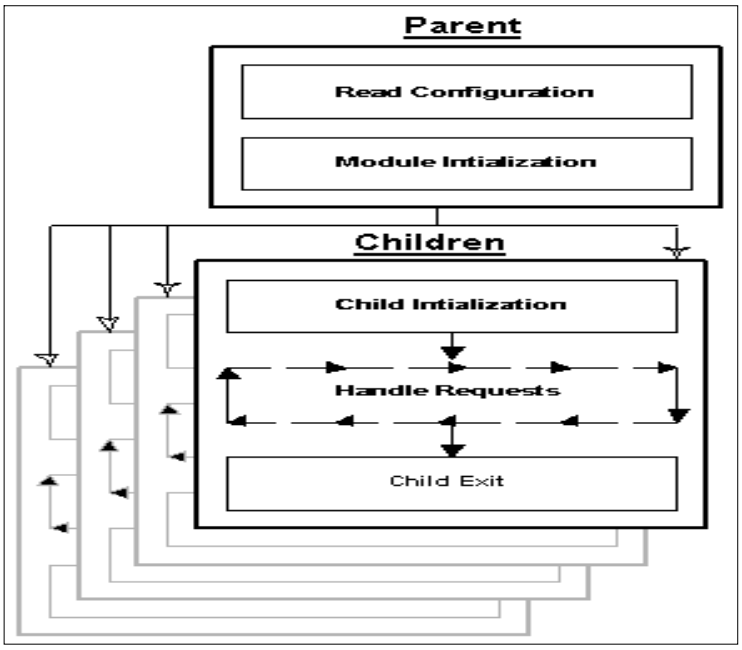

그림 3.3.2 아파치 프로세서 구성도

2) DBMS 부분

본 연구에서는 DBMS로 Mysql DBMS를 사용하였다.

3) 사용자 부분

사용자 부분은 크게 PC에서 Internet Browser를 사용하여 시스템에 접근하는 방법과 PDA를 사용하여 시스템에 접근하는 방법을 제공하고 있다. 이를 통하여 통신이 가능한 공간에서의 지리적 제약을 최소화하고 시스템의 사용률을 향상 시킬 수 있다.

나. Web Server 구축 및 테스트

전체적인 시스템의 OS는 Windows기반을 사용하였고 Web Server는 여러 분야에서 안정적인 운영이 검증된 Apache 2.0을 사용하였으며 WAS는 서버 부분 개발이 JAVA 기반으로 개발되어지므로 JSP 컴퍼일러를 내장하고 있는 Jakarta-Tomcat 5.5.9 버전을 사용하였다.

1) Web Server 구축

Web Server는 Apache 2.0을 사용하였으며 Tomcat과의 연동을 위하여 관련 Module을 설치하였다. Apache Web Server는 Apache group에서 다양한 OS에서 설치 가능하도록 제공하고 있다. 본 시스템에서는 Windows에서 설치 가능한 'apache_2.0.55-win32-x86-no_ssl.msi' 실행 파일을 다운 받아 설치하였다. Apache Web Server의 설치는 절차에 의하여 쉽게 설치가 가능하다.

그림 3.3.3 Apache 설치 첫 화면

2) Tomcat과의 연동을 위한 설정

Tomcat과의 연동을 위해서는 추가적인 설정 및 module 추가가 이루어 져야 한다.

가) module 추가

'jakarta-comcat-connectors-jk2.04-win32-apache2.0.49.zip' 을 Apache web site에서 Down 받아 'modules\mod_jk2.so' 파일을 Apache가 설치되어진 곳의 mo

dules 디렉토리에 복사한다.

나) workers 파일 설정

Tomcat과의 연동을 위해 Worker 파일을 아래와 같이 작성한다.

[shm]file=D:/Apache2/logs/shm.file

.size=1048576[channel.socket:localhost:8009]port=8009host=127.0.0.1[ajp13:localhost:8009]channel=channel.socket:localhost:8009[status:status][uri:localhost/jkstatus/*]group=status:status[uri:localhost/jsp-examples/*]worker=ajp13:localhost:8009[uri:localhost/servlets-examples/*]worker=ajp13:localhost:8009

다) httpd.conf 파일 수정

Tomcat과의 연동을 위해 Apache Web Server의 설정을 저장한 파일인 httpd.conf 파일을 아래와 같이 수정한다.

LoadModule jk2_module modules/mod_jk2.soJkSet config.file "c:/Apache2/conf/workers2.properties" --> 다른 디렉토리일 경우 꼭 변경해주어야 한다.

라) 테스트

Web Server의 설치된 후 정상적인 설치 확인을 위해 Web Server를 실행한 후에 직접 접속하여 설치를 확인하였다. 정상적으로 실행이 되면서 아래와 같이 첫 페이지를 확인할 수 있으면 Apache Web Server는 정상적으로 설치되었다 할 수 있다.

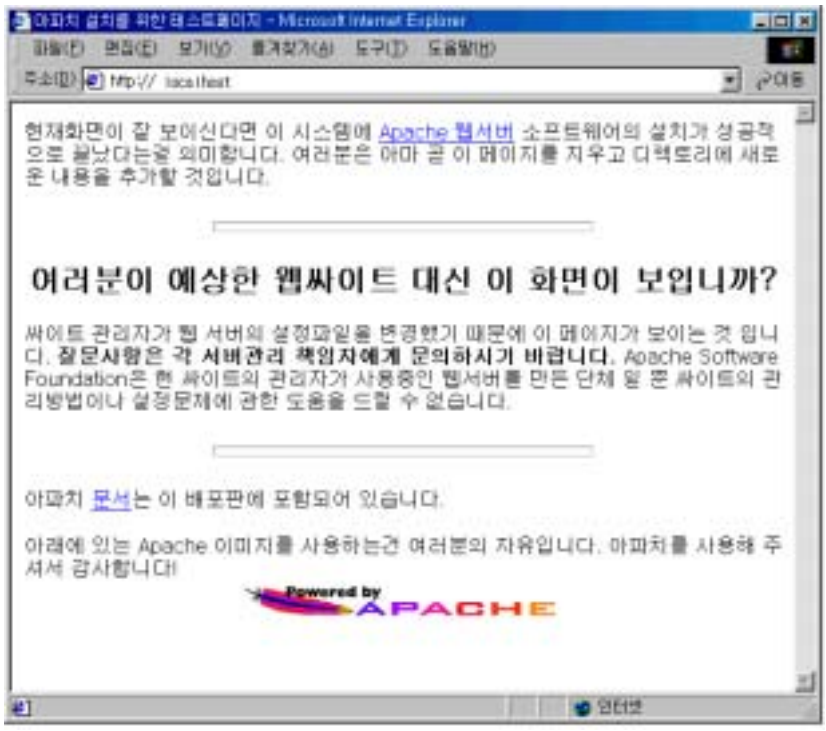

그림 3.3.4 Apache 설치 Test 화면

다. WAS 구축 및 테스트

본 연구에서는 Server 부분에 대한 개발을 JAVA 기반으로 하였고 이를 위해 WAS Server로는 Tomcat을 사용하였으며 Web Server로는 Apache를 사용하였다.

1) Tomcat을 단독으로 사용하는 데에는 다음과 같은 문제점이 있으므로 본 연구에서는 Web Server인 아파치와 같이 사용하였다.

가) 아파치 웹서버에 비해 정적 페이지 서비스가 느리다.

나) 아파치만큼 다양한 웹서버 환경을 제공하지 않는다.

다) 아파치 만큼 웹서버로서 안정적이지 않다.

라) 많은 사이트들이 기존의 웹서버 환경에서 서비스하고 있는 것들이 많다.

마) 웹서버의 기본은 클라이언트의 HTTP(Hypertext Transfer Protocol) 요청을 기다리다가 요청이 들어왔을 때 필요한 컨텐트를 제공함으로써 이 요청을 서비스한다. 서블릿 컨테이너를 추가함으로써 웹서버는 다음과 같은 행동을 한

다.

바) 서블릿 요청이 있기 전에 서블릿 컨테이너 어댑터 라이브러리를 로딩하고 초기화한다.

사) 요청이 들어오면 서블릿에 대한 요청인지를 체크해서, 어댑터가 이 요청을 받아 처리한다. 다시 말해 어댑터는 HTTP(Hypertext Transfer Protocol) 요청중에 서블릿에 대한 요청을 구분해내야 하는데, 일반적으로 요청 URL 패턴에 근거한다.

아) 톰캣이 가동되면 자동적으로 아파치에 대한 환경 파일(TOMCAT_HOME/conf/tomcat- apache.conf)을 생성한다. 본 연구에서는 httpd.conf 파일에 'include TOMCAT_HOME/conf/ tomcat-apache.conf'를 추가시켜 아파치는 Tomcat의 클라이언트 모듈(웹서버 어뎁터)을 로딩 하도록 환경을 구성하였다.

2) Tomcat의 성능향상을 위한 설정

JVM(Java Virtual Machine)의 메모리 변경: 일반적으로 JVM(Java Virtual Machine)은 Java heap과 JVM 자체를 위한 초기 사이즈를 할당한다. 본 연구에서는 Tomcat의 성능 향상을 위해서 JVM의 메모리를 추가하였다.

가) JVM의 Thread 환경 변경

Java2 SDK부터는 native와 green Threading을 지원하는데, 일반적으로 native Thread가 I/O관련 애프리케이션과 관련해서는 성능을 향상시켜주고, green Thread JVM은 서버에 스트레스를 덜 주는 것으로 알려져 있다. 본 연구에서는 실시간으로 데이터의 전달이 이루어져야 함으로 서버에 스트레스가 되더라도 native Thread로 설정하였다. Tomcat의 server.xml 환경 설정 파일에 HTTP(Hypertext Transfer Protocol)와 AJPV12에 대한 두 가지 Connector 설정이 되어있다. Apache Web Server와 같이 외부 웹서버에 Tomcat을 서블릿 컨테이너로 통합해 사용할 경우 HTTP 요청을 Tomcat이 처리할 필요가 없으므로 본 연구에서는 8080 포트에 대한 Connector는 Comment 처리하여 Tomcat 가동 시 불필요한 Connector가 기동되지 않도록 하였다.

Tomcat은 멀티 Thread를 지원하는 서블릿 컨테이너이므로 각각의 요청은 Thread에 의해 실행된다. 디폴트로 하나의 요청이 들어오면 Tomcat은 새로운 Th

read를 생성해서 해당 요청을 처리하도록 한다.

나) 문제점

- 매 번 요청 시에 Thread를 실행시키고 중단시키는 것은 OS와 JVM에게 불필요한 부담을 준다.
- 자원의 사용을 제한하기가 어렵다.

다) 해결 방법

- Thread Pool을 사용
- Thread Pool을 사용하는 경우 요청에 대해 새로운 Thread를 생성하지 않고 Thread Pool에서 Thread를 가져다 사용하고, 서비스가 끝나면 다시 반납.
- Thread를 재사용함으로써 생성하고 제거하는 소모적인 일을 하지 않아도 된다
- 동시에 사용되는 Thread의 갯수를 제한할 수도 있다.

라) 본 연구에서의 Thread 환경설정(조건)

- 동시 사용자 수 : 300명
- 동시에 서비스 할 수 있는 최대 Thread 갯수 : 300
- ideal 시점에 유지하는 최대 Thread 갯수 : 200
- 최소 Thread 유지 개수 : 50

마) 환경 설정

Tomcat에 대한 환경 설정은 다음과 같다.

```
<Connector className
    = "org.apache.tomcat.service.SimpleTcpConnector">
    <Parameter name="handler"              value="org.apache.tomcat.service.co
nnector.Ajp12ConnectionHandler"/>
    <Parameter name="port" value="8007"/>
```

```
    <Parameter name="max_Threads" value="300"/>
    <Parameter name="max_Spare_Threads" value="200"/>
    <Parameter name="min_Spare_Threads" value="50" />
</Connector>
```

바) Apache와 연동

Tomcat 5.x 의경우 <Connector>에서 설정한 값에 의해 GET 방식의 인코딩이 결정된다. 다음처럼 por="8009"의 커넥트 설정 부분에 URIEncoding="euc-kr" 설정을 추가해야 한다.

```
<Connector port="8009"redirectPort="8443" debug="0"URIEncoding="euc-kr
"/>
```

사) 설치 테스트

Tomcat 설치를 확인하기 위해서는 우선 Tomcat을 단독으로 실행시켜 정상적으로 첫 페이지가 나타나는지를 확인하고 다음으로 apache Web Server를 같이 실행시켜 정상적으로 첫 페이지 나타나는가를 확인한다.

- 단독 실행 시 테스트 화면: Tomcat을 단독으로 실행하고 'HTTP://localhost:8080"과 Tomcat의 기본 포트 번호를 함께 사용하여 테스트를 한다. 아래와 같은 화면이 나타나면 정상적인 작동 상태라 할 수 있다.

그림 3.3.5 Tomcat 단독 실행 테스트 화면

- Apache와 Tomcat을 연동한 테스트: tomcat과 Apache를 같이 실행 한 후에 정상적으로 작동 되는 가를 확인한다. 'HTTP://localhost'와 같이 url을 입력하였을 때 아래 그림과 같은 화면이 나타나면 정상적으로 작동이 되는 것이다. 이는 Tomcat을 단독으로 실행 했을 때의 화면과 동일하다. 따라서 Apache는 8009 포트를 사용하여 80 포트를 통한 요청 즉 Web 요청에 대한 처리를 Tomcat으로 전송함을 알 수 있다.

그림 3.3.6 Tomcat과 Apache 연동을 통한 테스트 화면

4. 서버 프로그램 설계 및 구현

본 연구에서는 MVC Model 기반의 Struts를 개선하여 개발 Framework으로 사용하였다. 이를 통하여 Business Logic을 완전히 분리하여 개발하였으며 컨트롤의 흐름을 명확히 하였다. 또한 WebService를 위해 공개되어지는 부분은 Business Logic이 전혀 포함되어지지 않게 하고 서버와 클라이언트간의 통신만을 담당하도록 하여 WebService를 위해 별도의 Business Logic을 구성하지 않고 기존의 Business Logic을 사용하도록 하였다.

가. Struts

웹애플리케이셔을 개발할 때 개발자들이 사용하는 방식 중에 모델1 방식과 모델2 방식이 있으며, 그 중에서도 모델1 방식을 가장 많이 사용하고 있으며, 또한 가장 쉽기도 하다.

1) 모델1 개발방식

모델1과 모델2의 가장 큰 차이점은 클라이언트의 요청이 진입하는 지점이 다르다는 것이다. 모델1에서의 클라이언트의 요청을 처리하는 부분은 JSP이다.

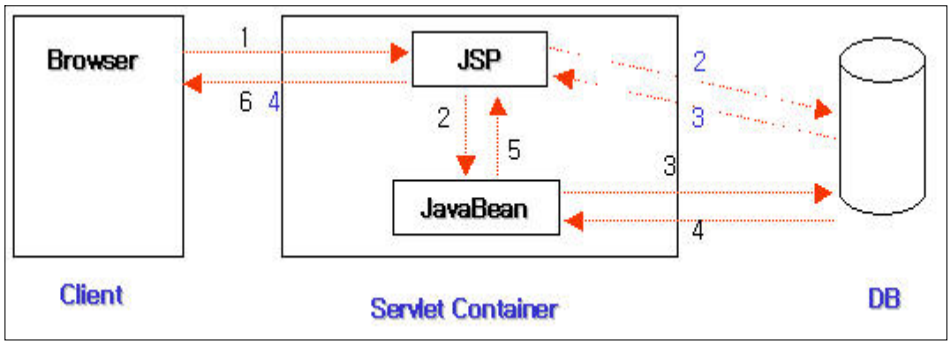

그림 3.4.1 모델 1의 구조

모델 1 구조의 장단점은 아래와 같다.

가) 장점

- 개발 시간이 단축된다.
- 단순한 페이지 흐름으로 인해 MVC 구조에 대한 추가적인 교육의 필요성이 없다.

나) 단점

- HTML , 자바스트립트, JSP로직이 한 페이지에 쓰여지므로 어플리케이션이 복잡해질수록 유지보수가 힘들다.
- 디자이너와 개발자간의 원활한 의사소통이 필요하다.

2) 모델2 개발방식

모델 2개발 방식과 모델 1개발 방식의 가장 중요한 차이점은 클라이언트의 요청이 진입하는 지점이 다르다는 것이다. 모델 1의 초기 진입 지점은 JSP가 담당하였지만 모델 2의 진입 지점은 컨트롤러 부분을 따로 두어 처리한다. 대부분 웹 어플리케이션의 컨트롤러는 서블릿이 담당하게 된다. 아래 그림은 모델 2방식으로 클라이언트의 요청을 처리하는 과정을 도식화한 것이다.

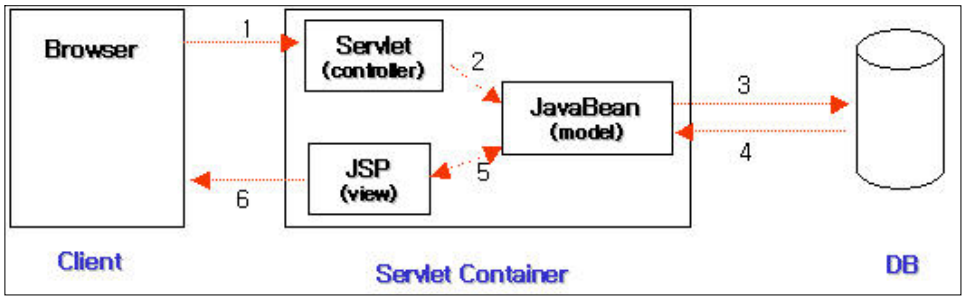

그림 3.4.2 모델 2의 구조

모델2 방식의 장단점은 아래와 같다.

가) 장점

- 로직과 프리젠테이션의 분리로 인해 어플리케이션이 명료해지며 유지보수와 확장이 용이하다.
- 디자이너와의 작업이 분리되어져 책임의 소재가 명확하다.

나) 단점

- 개발 초기에 아키텍쳐 디자인을 위한 시간의 소요로 개발 기간이 증가한다.
- MVC 구조에 대한 개발자들의 이해가 필요하다.
- 위와 같이 컨트롤러,모델,뷰로 구분하여 개발하는 방법을 모델 - 뷰 - 컨트롤러(MVC) 패턴이라 한다. 모델 2는 MVC 패턴을 바탕으로 개발하는 것이다.

다) 클라이언트의 요청이 모델 2로 처리되는 세부 과정

- 모델 2에서의 모든 요청은 제일 먼저 컨트롤러인 서브릿으로 진입하게 된다.
- 컨트롤러는 클라이언트의 요청을 받아 모델에 작업을 위임한다.
- 컨트롤러가 모델에 작업을 위임할 때 클라이언트로부터 전달된 데이터를 같이 전달하게 된다.
- 모델이 작업을 완료한 다음 컨트롤러에 결과물을 반환한다.
- 컨트롤러는 모델로부터 반환된 결과물의 상태와 클라이언트가 전달한 인자에 따라 클라이언트에게 보여줄 뷰를 결정한다. 호출되는 뷰에는 모델로부터 반환된 결과물을 같이 전달한다.

● 뷰는 최종적으로 컨트롤러로부터 전달된 결과물을 클라이언트에게 보여준다.

3) 스트러츠 프레임워크 개요

모델2로 개발하려는 시도가 점차 증가하고 있지만 재 사용성의 부족이라는 한계를 느끼고 많은 개발자들이 모델2를 포기하는 경우가 종종 있다. 그러다 재사용성을 강화하여 모델2에 기반한 Framework가 등장하게 되었으며, 그 중에 하나가 아파치 그룹에서 개발한 Struts다. 웹 애플리케이션을 만들기 위해 필요한 많은 부분들을 미리 만들어 놓았기 때문에, 서블릿과 JSP 기반 하에서 개발하는 웹 애플리케이션의 개발 기간을 상당부분 단축시키는 효과를 가져올 수 있다.

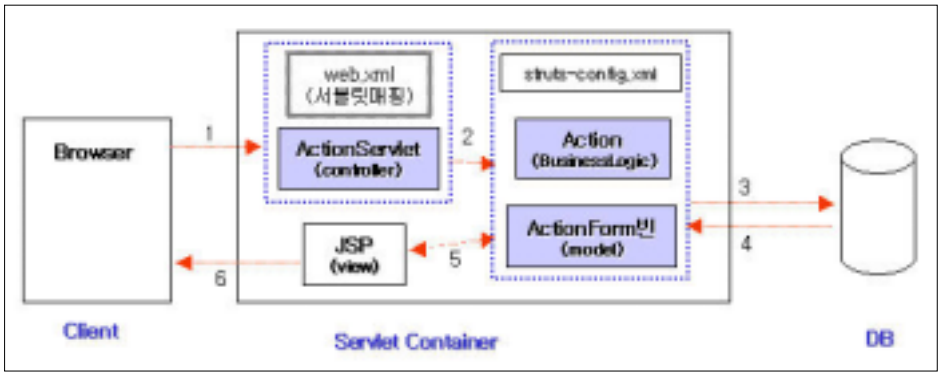

그림 3.4.3 Struts 의 구조

우리가 하나하나 새로 만들어야 했던 컨트롤러나, 모델 쪽의 기본 패턴들이 이미 만들어져 있어 그것을 사용하여 시스템에 맞게 Overriding하면 된다.

가) Struts의 구조

모델 2에서 Struts로 발전하면서 크게 달라진 부분은 UserCommandFactory 부분이다. 모델 2의 UserCommandFactory에는 수많은 if/else절이 존재했다. 프로젝트가 커지면 커질수록 더 많은 if/else를 가지는 구조로 작성되는데 Struts에서는 이 부분을 XML로 해결했다. Struts는 JSP와 Action 클래스 사이의 매핑을 소스가 아닌 XML을 이용하여 해결하고 있다. 그렇기 때문에 새로운 Action클래스가 추가될 경우, 소스의 수정 없이 XML 파일에 JSP와 Action클래스를 매핑해

주면 된다. 아래 그림은 Struts를 이용하여 클라이언트의 요청을 처리하는 과정을 보여주고 있다. 여기서 보여주는 클래스 이외에도 많은 클래스들이 웹 어플리케이션 개발을 위하여 존재 하지만 Struts에서 하나의 요청을 처리하기 위하여 중심이 되는 클래스들만 보여주고 있다.

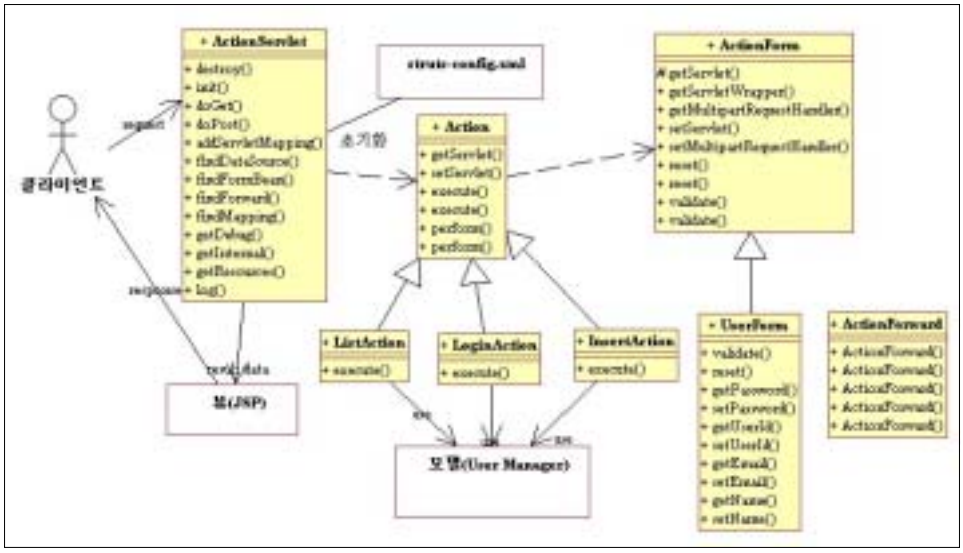

그림 3.4.4 Struts 클래스 다이어그램

위의 그림에서 모델 2와 비교하여 가장 큰 차이점은 UserCommandFactory가 없어졌다는 것이다. Struts에서 UserCommandFactory 역할을 하는 부분은 struts-config.xml이다. 이 XML파일이 JSP와 Action클래스를 매핑하는 역할을 한다. 두 번째 큰 차이점은 ActionForm클래스가 추가된 것이다.

나) Struts Framework의 장단점

Struts는 모델 2 방식으로 웹 애플리케이션을 개발하고자 하는 개발자들에게 많은 편의성을 제공한다. 모델 2로 개발하기 위하여 필요한 많은 부분들을 미리 개발해 놓았기 때문에 개발시간 또한 상당히 단축시킬 수 있는 효과를 가져올 수 있다. Struts는 이 절에서 살펴본 항목 이외에도 상당히 많은 부분을 재사용 가능하도록 이미 구현해 놓은 상태이다. 따라서 Struts를 정확히 이해하고 제대로 사용할 수만 있다면 아무리 큰 웹 애플리케이션일지라도 개발이 가능하다. 하

지만 Struts를 제대로 사용하기 위해서는 상당한 학습시간을 필요로 한다. 모델2로 발상의 전환이 있어야할 뿐만 아니라 JSP와 Action클래스 매핑을 전담하는 struts-config.xml에 대한 이해까지 선행되어야 한다.

대부분의 개발자들이 Struts에 대한 이해도가 높다면 Struts를 이용하여 웹 애플리케이션을 개발하는 기간이 오히려 모델1 방식으로 개발할 때보다 더 단축할 수 있는 효과를 가져올 수 있다.

나. Struts기반의 Framework 설계 및 구현

1) 클래스 구조

시스템 개발에 앞서 본 연구에서 사용되어질 클래스들의 namespace를 정의 하고자 한다. 클래스의 namespace는 일반적인 자바 개발 naming rule을 사용하여 'com.acai.*로 하였다. 본 연구에서는 데이터베이스의 접근을 전담하는 XXXDAO 클래스와 데이터베이스의 접근을 관리하는 XXXManager 클래스로 데이터베이스와의 접근을 관리하였다. 클래스는 기능에 따라 분류하였으며 아래와 같다.

가) com.acai.utils: 시스템에서 사용되어지는 기본 기능 클래스를 정의하고 있다.

- DateTime: 시간과 날짜에 관한 기능을 제공하고 있다.
- MessageUtil: 에러메세지에 관한 기능을 정의하고 있다.
- PropertyUtil: 속성에 관련된 기능을 정의 하고 있다.

나) com.acai.db: 데이터베이스에 관련된 클래스를 정의하고 있다.

- ConnectionManager: 데이터베이스와의 연결을 관리하는 클래스이다. 본 연구에서는 Connection Pool을 사용하여 데이터베이스 연동 속도 및 성능을 향상시켰으며 데이터베이스의 변경 등 데이터베이스와의 연결에 관련된 변경 사항은 ConnectionManager 클래스의 변경만으로 적용이 가능하다.

다) com.acai.model: 데이터베이스의 business logic을 정의하고 있다.

- UtilManager: 본 시스템에서 사용되어지는 코드의 관리를 위해 사용되는 관리 클래스로 데이터베이스와 연결이 이루어지는 부분이다.

- UtilDAO: 본 시스템에서 사용되어지는 코드 관리를 위한 Business Logic이 정의되어진 클래스이다.
- UserNotFoundException: 요청 사용자가 존재하지 않을 경우 반환되는 Exception 클래스이다.
- UserNotConfirmException: 잘못된 사용자 등록 확인이 되었을 경우 반환되는 Exception이다.
- UserManager: 사용자 등록 및 변경 삭제 등 사용자 관리에 사용되는 관리 클래스로 데이터베이스와 연결이 이루어지는 부분이다.
- UserDAO: 사용자 등록 및 변경 삭제 등 사용자 관리 Business Logic이 정의되어진 클래스이다.
- ProductManager: 접수관리, 농산물정보 변경 삭제 등록 등 접수와 관련된 전반적인 관리를 위해 사용되어지는 클래이스로 데이터베이스와 연결이 이루어지는 부분이다.
- ProductDAO: 접수관리, 농산물정보 변경 삭제 등록 등 접수와 관련된 전반적인 관리 Business Logic이 정의 되어진 클래스이다.

라) com.acai.resources: 데이터베이스 연결을 위한 속성을 정의하고 있다.

- ApplicationResources_ko.properties: Exception 메시지에 대한 정의가 되어진 속성 파일이다.

마) com.acai.service: Webservice를 위한 클래스를 정의하고 있다.

- UserAuthImpl: UserAuthIF에 정의되어진 메소드에 대한 실행 Logic이 정의되어져 있다. UserAuthImpl에는 Business Logic이 정의 되어져 있지 않으며 단지 원격객체 참조와 내부 Business Logic간의 연결을 위한 Logic만이 존재한다. 따라서 본 시스템은 동일한 Business Logic을 통해 PDA와 Internet Browser가 작동한다.
- UserAuthIF: Axis를 이용한 WebService를 위해 정의되어진 인터페이스로 이 부분이 WSDL로 변환되어져 원격객체 참조의 대상이 된다. PDA에서 원격객체를 참조 할 경우 UserAuthIF 인터페이스에 정의되어진 메소드만을 사용할 수 있다.

바) com.acai.unit: 시스템에서 사용되어지는 객체에 대한 클래스를 정의하고 있다.

- Zipcode: 주소에 대한 객체 정의 클래스
- UserE: 추가 사용자 정보에 대한 객체 정의 클래스
- User: 사용자 정보에 대한 객체 정의 클래스
- ReceiptServ: WebService에서 사용되어지는 접수증 객체에 대한 정의 클래스
- ReceiptProduct: 접수증의 농산물 등록에 관한 정의 클래스
- ReceiptMain: 접수증의 사용자 정보에 관한 정의 클래스
- ReceiptAuction: 접수증의 추가 정보에 관한 정의 클래스
- Receipt: 접수증에 대한 객체 정의 클래스
- Receipt2: Webservice에서 사용되어지는 접수증에 대한 객체 정의 클래스
- RceiptProductServ: Webservice에서 사용되어지는접수증의 농산물 등록에 관한 정의 클래스
- Pummok: 품목 코드에 관한 정의 클래스
- Nonghup: 농협 코드에 관한 정의 클래스
- Item: 품종 코드에 관한 정의 클래스
- Grade: 등급 코드에 관한 정의 클래스
- CommManage: 일반 코드 관리에 관한 정의 클래스
- ChangedItem: 농산물 변경 관리에 관한 정의 클래스
- BasicStatic: 기본 통계정보에 관한 정의 클래스

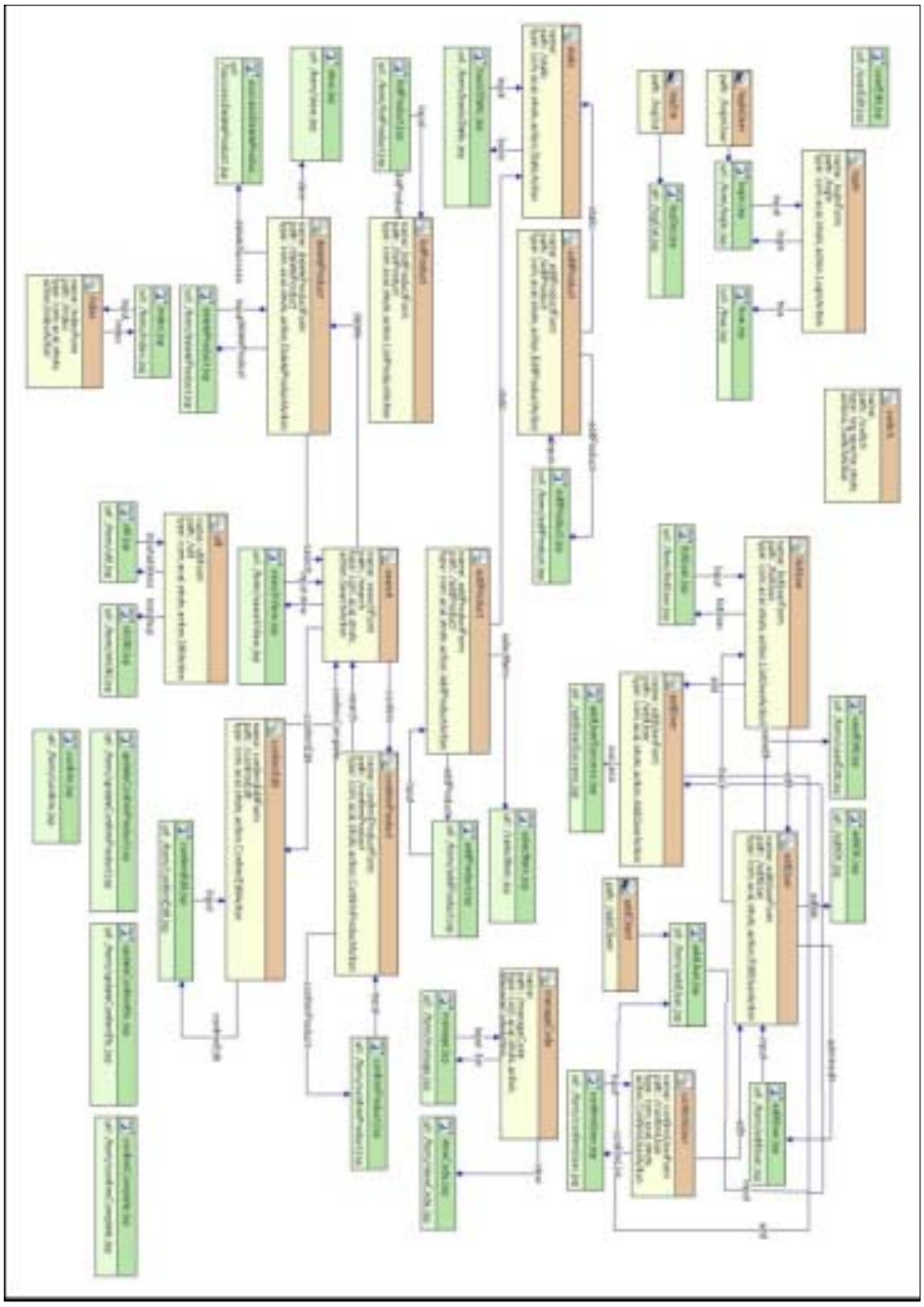

그림 3.4.5 Struts 구성 화면

2) struts-config.xml 파일의 구성

Struts에서는 모델 2와 비교하여 가자 큰 차이점으로 UserCommandFactory가 없어졌다는 것이다. Struts에서 UserCommandFactory 역할을 하는 부분은 struts-config.xml이며 이 XML파일이 JSP와 Action클래스를 매핑하는 역할을 한다. 아래는 본 시스템을 구성하기 위한 struts-config.xml 파일의 내용을 나타내고 있다.

```
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 1.1//EN" "http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd">

<struts-config>
  <data-sources />
  <form-beans >
    <form-bean  name="loginForm"  type="com.acai.struts.form.LoginForm" />
    <form-bean  name="addUserForm"  type="com.acai.struts.form.AddUserForm" />
    <form-bean  name="addProductForm"  type="com.acai.struts.form.AddProductForm" />
    <form-bean  name="editProductForm"  type="com.acai.struts.form.EditProductForm" />
    <form-bean  name="listProductForm"  type="com.acai.struts.form.ListProductForm" />
    <form-bean  name="deleteProductForm"  type="com.acai.struts.form.DeleteProductForm" />
    <form-bean  name="listUserForm"  type="com.acai.struts.form.ListUserForm" />
    <form-bean  name="editUserForm"  type="com.acai.struts.form.EditUserForm" />
```

```
    <form-bean name="confirmProductForm" type="com.acai.struts.form.ConfirmProductForm" />
    <form-bean name="indexForm" type="com.acai.struts.form.IndexForm" />
    <form-bean name="utilForm" type="com.acai.struts.form.UtilForm" />
    <form-bean name="confirmUserForm" type="com.acai.struts.form.ConfirmUserForm" />
    <form-bean name="ssForm" type="com.acai.struts.form.SsForm" />
    <form-bean name="confirmEditForm" type="com.acai.struts.form.ConfirmEditForm" />
    <form-bean name="searchForm" type="com.acai.struts.form.SearchForm" />
  </form-beans>

  <global-exceptions />
  <global-forwards />


  <action-mappings >

    <action
      attribute="loginForm"
      input="/form/login.jsp"
      name="loginForm"
      path="/login"
      scope="request"
      validate="true"
      type="com.acai.struts.action.LoginAction">
      <forward name="login" path="/form/login.jsp" />
      <forward name="true" path="/true.jsp" />
```

```
</action>

<action
  attribute="addUserForm"
  input="/form/addUser.jsp"
  name="addUserForm"
  path="/addUser"
  scope="request"
  type="com.acai.struts.action.AddUserAction">
  <forward name="add" path="/form/addUser.jsp" />
  <forward name="success" path="/addUserSuccess.jsp" />
</action>

<action
  attribute="addProductForm"
  input="/form/addProduct.jsp"
  name="addProductForm"
  path="/addProduct"
  scope="request"
  type="com.acai.struts.action.AddProductAction">
  <forward name="selectItem" path="/selectItem.jsp" />
  <forward name="addProduct" path="/form/addProduct.jsp" />
  <forward name="static" path="/static.do" />
</action>

<action
  attribute="editProductForm"
  input="/form/editProduct.jsp"
  name="editProductForm"
  path="/editProduct"
```

```
    scope="request"
    type="com.acai.struts.action.EditProductAction" >
    <forward name="editProduct" path="/form/editProduct.jsp" />
    <forward name="static" path="/static.do" />
</action>

<action
    attribute="listProductForm"
    input="/form/listProduct.jsp"
    name="listProductForm"
    path="/listProduct"
    scope="request"
    type="com.acai.struts.action.ListProductAction" >

    <forward name="listProduct" path="/form/listProduct.jsp" />
</action>

<action
    attribute="deleteProductForm"
    input="/form/deleteProduct.jsp"
    name="deleteProductForm"
    path="/deleteProduct"
    scope="request"
    type="com.acai.struts.action.DeleteProductAction" >
    <forward name="deleteProduct" path="/form/deleteProduct.jsp" />
    <forward name="deleteSuccess" path="/successDeleteProduct.jsp" />
    <forward name="view" path="/form/view.jsp" />
    <forward name="search" path="/search.do" />
</action>
```

```
<action
  attribute="listUserForm"
  input="/form/listUser.jsp"
  name="listUserForm"
  path="/listUser"
  scope="request"
  type="com.acai.struts.action.ListUserAction" >
  <forward name="edit" path="/editUser.do" />
  <forward name="listUser" path="/form/listUser.jsp" />
  <forward name="add" path="/addUser.do" />
</action>

<action
  attribute="editUserForm"
  input="/form/editUser.jsp"
  name="editUserForm"
  path="/editUser"
  scope="request"
  type="com.acai.struts.action.EditUserAction" >

  <forward name="back" path="/listUser.do" />
  <forward name="editok" path="/editOK.jsp" />
  <forward name="useredit" path="/form/userEdit.jsp" />
  <forward name="adminedit" path="/form/editUser.jsp" />
</action>

<action
  attribute="confirmProductForm"
  input="/form/confirmProduct.jsp"
  name="confirmProductForm"
```

```
        path="/confirmProduct"
        scope="request"
        type="com.acai.struts.action.ConfirmProductAction">
        <forward  name="confirmProduct"  path="/form/confirmProduct.jsp"
/>
        <forward name="search" path="/search.do" />
      </action>

      <action path="/switch"
         type="org.apache.struts.actions.SwitchAction"
         validate="false"/>
      <action
        attribute="indexForm"
        input="/form/index.jsp"
        name="indexForm"
        path="/index"
        scope="request"
        type="com.acai.struts.action.IndexAction" >
        <forward name="index" path="/form/index.jsp" />
      </action>

      <action forward="/form/addUser.jsp" path="/addClient" />
      <action forward="/form/login.jsp" path="/loginUser" />
      <action forward="/logOut.jsp" path="/logOut" />

      <action
        attribute="utilForm"
        input="/form/util.jsp"
        name="utilForm"
        path="/util"
```

```
    scope="request"
    type="com.acai.struts.action.UtilAction" >
    <forward name="address" path="/form/util.jsp" />
    <forward name="nonghup" path="/form/nhUtil.jsp" />
</action>

<action
    attribute="confirmUserForm"
    input="/form/confirmUser.jsp"
    name="confirmUserForm"
    path="/confirmUser"
    scope="request"
    type="com.acai.struts.action.ConfirmUserAction" >
    <forward name="confirmList" path="/form/confirmUser.jsp" />
    <forward name="edit" path="/editUser.do" />
</action>

<action
    attribute="confirmEditForm"
    input="/form/confirmEdit.jsp"
    name="confirmEditForm"
    path="/confirmEdit"
    scope="request"
    type="com.acai.struts.action.ConfirmEditAction" >
    <forward name="confirmEdit" path="/form/confirmEdit.jsp" />
    <forward name="confirmComplete" path="/search.do" />
</action>

<action
    attribute="searchForm"
```

```
        input="/form/searchView.jsp"
        name="searchForm"
        path="/search"
        scope="request"
        type="com.acai.struts.action.SearchAction">
        <forward name="view" path="/form/searchView.jsp" />
        <forward name="confirm" path="/confirmProduct.do" />
        <forward name="delete" path="/deleteProduct.do" />
        <forward name="confirmEdit" path="/confirmEdit.do" />
      </action>

      <action
        input="/form/manage.jsp"
        path="/manageCode"
        type="com.acai.struts.action.ManageCodeAction">
        <forward name="list" path="/form/manage.jsp" />
        <forward name="view" path="/form/viewCode.jsp" />
      </action>

      <action
        input="/form/basicStatic.jsp"
        path="/static"
        type="com.acai.struts.action.StaticAction">
        <forward name="basic" path="/form/basicStatic.jsp" />
      </action>

    </action-mappings>
      <controller
contentType="text/html;charset=euc-kr"
debug="3"
```

```
locale="true"
nocache="true"
processorClass="com.acai.processor.MyAppRequestProcessor"/>

  <message-resources
    parameter="com.acai.resources.ApplicationResources"/>

  <plug-in  className="org.apache.struts.validator.ValidatorPlugIn">
        <set-property
        property="pathnames"
        value="/WEB-INF/validator-rules.xml,/WEB-INF/validation.xml"/>
</plug-in>

</struts-config>
```

3) 서버 Application 구성

Struts 형태에서는 Servlet을 통해 제어가 이루어지며 이동은 Mapping 정보에 따라서 이루어진다. 농산물 상장 등록 시스템의 각 부분별 Struts 구조는 아래와 같다.

가) Login/ Logout

Login과 Logout은 아래의 그림과 같이 LoginAction Servlet을 통해 제어가 이루어진다. 입력으로는 login.jsp가 사용되어지며 login 화면은 login.jsp이며 login이 성공적으로 이루어지면 true.jsp를 통해 login 성공을 알려 준다. Logout은 logOut servlet을 통해 제어되며 출력은 logOut.jsp를 통해 보여준다.

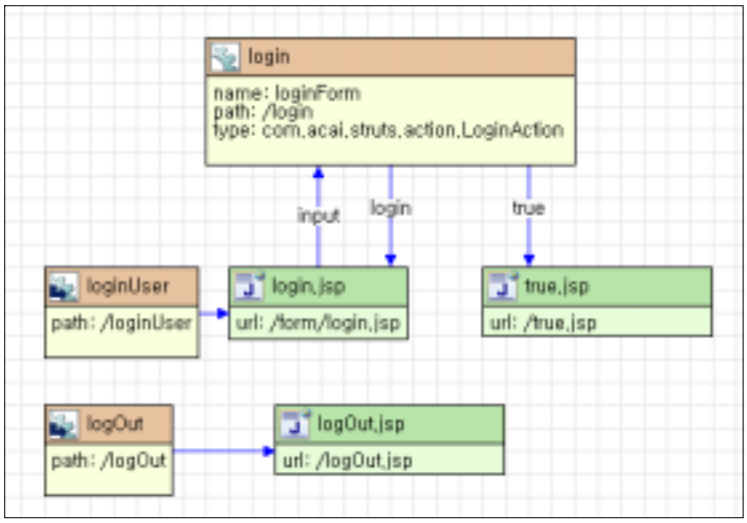

그림 3.4.6 Login

나) 사용자 관리

사용자 관리에 사용되어지는 servlet은 listUserAction, editUserAction, addUserAction, confirmUserAction 등이다. listUserAction는 현재 사용자의 리스트를 보여주며 editUserAction는 사용자 정보 수정을 담당하며 addUserAction는 사용자 추가, confirmUserAction는 등록요청 사용자에 대한 확인을 담당한다.

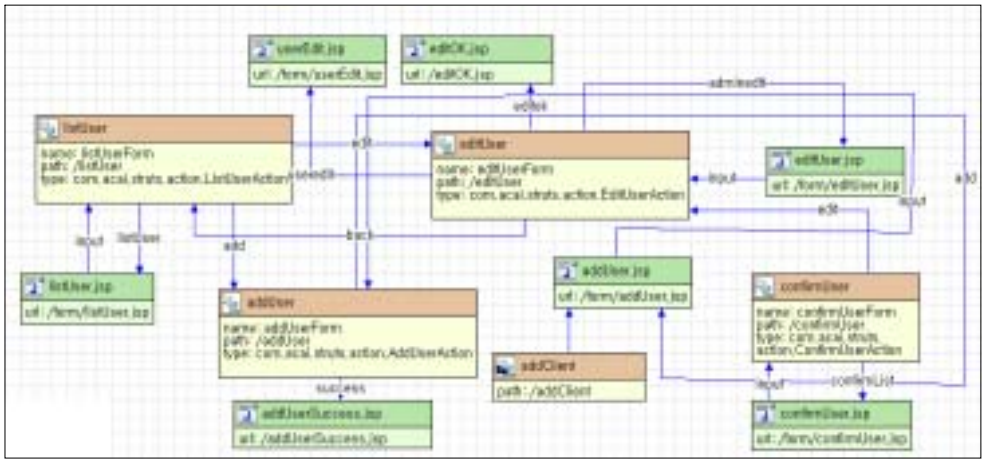

그림 3.4.7 사용자 관리

다) 기능 관리

기능 관리 Servlet은 UtilAction을 통해 제어되며 시스템에서 사용되어지는 함수를 정의하고 있다.

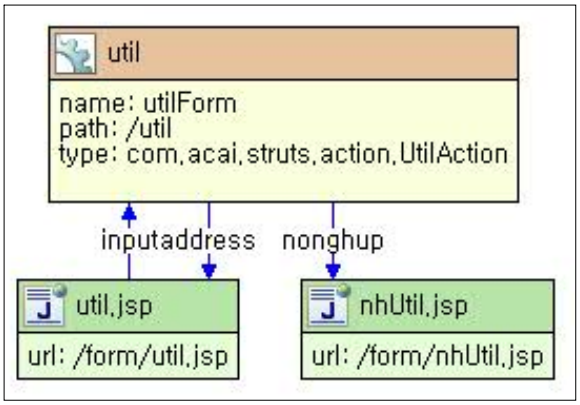

그림 3.4.8 기능 관리

라) 첫 화면 관리

첫 화면은 index.jsp를 통해 시작하며 indexAction servlet을 통해 관리가 이루어진다.

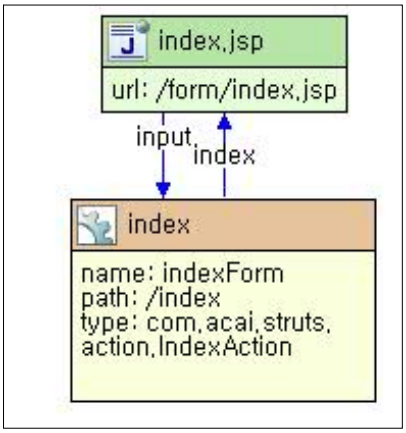

그림 3.4.9 첫 화면 관리

마) 농산물정보 수정

등록되어진 농산물 정보를 수정한다. EditProductAction Servlet을 통해 제어가 이루어지며 출력은 editProduct.jsp를 통해 이루어진다.

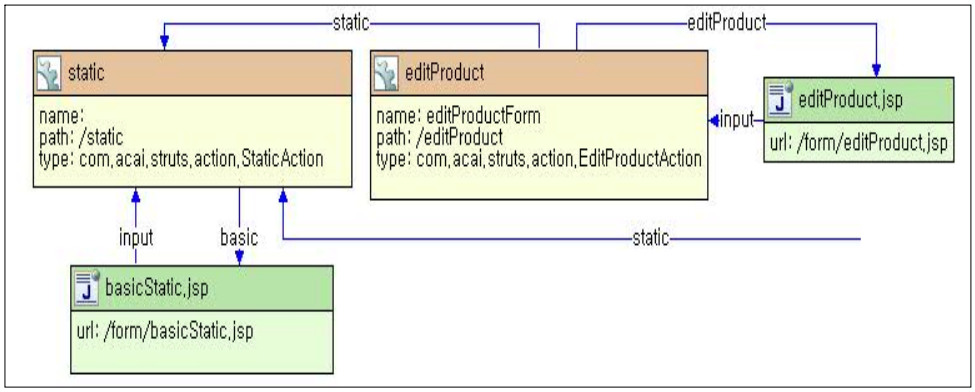

그림 3.4.10 농산물정보 수정

바) 제품목록 보기와 제품 삭제

제품목록 보기는 ListProductAction servlet을 통해 제어가 되어지며 입출력은 listProduct.jsp를 통해 보여 진다.

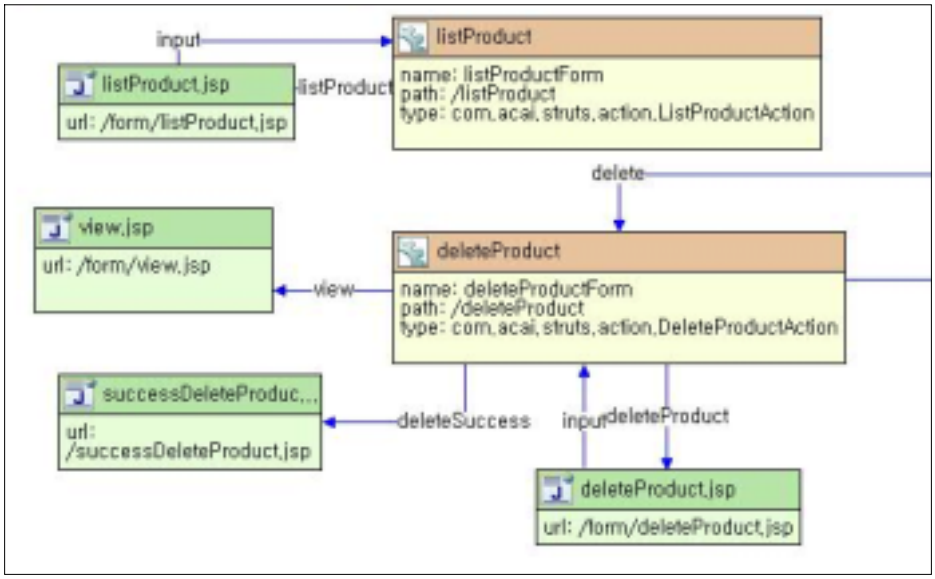

그림 3.4.11 제품목록 보기와 제품 삭제

사) 농산물 추가

신규 농산물 등록을 담당한다. AddProductAction을 통해 제어 되어지며 addProduct.jsp를 통해 입출력을 보여준다.

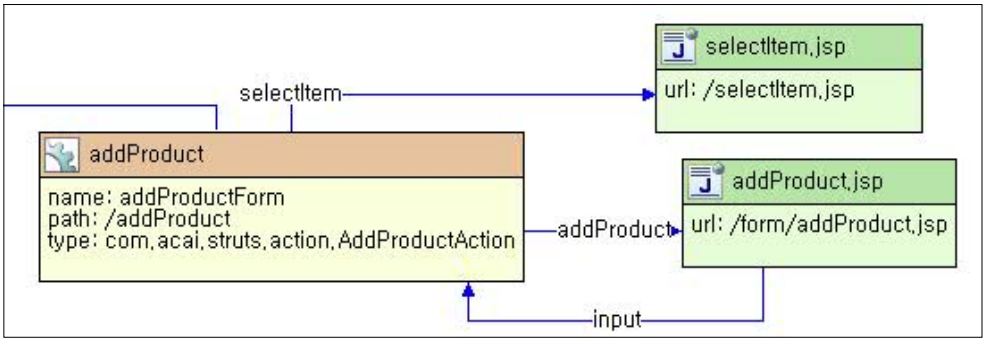

그림 3.4.12 농산물 추가

아) 입하확인

입하 확인을 담당한다. ConfirmProductAction은 입하 준비 중인 농산물의 리스트를 보여주며 confirmProduct.jsp를 통해 입출력된다. 입하확인은 ConfirmEditAction에서 담당하며 작업자에 의해 입하 품목의 수정도 가능하다. confirmEdit.jsp를 통해 입출력이 이루어진다.

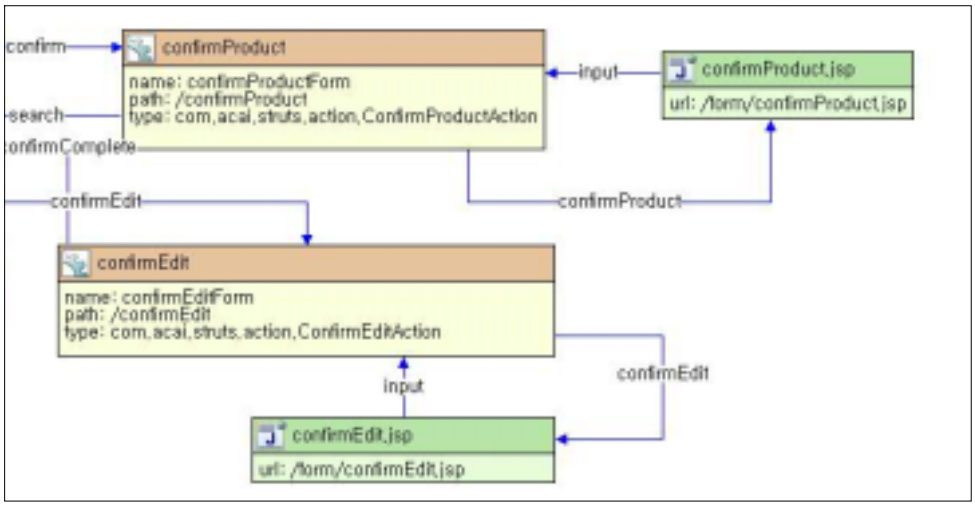

그림 3.4.13 입하 확인

자) 검색

검색은 SearchAction을 통해 제어되어진다. 입출력은 searchView.jsp를 통해 입출력이 이루어진다.

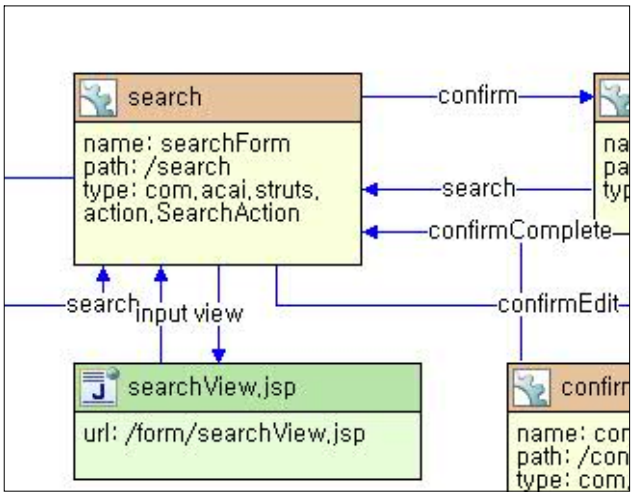

그림 3.4.14 검색

차) 코드 관리

시스템에서 사용되어지는 코드를 관리한다. ManageCodeAction이 제어를 담

당하며 manage,jsp를 통해 입출력되어진다.

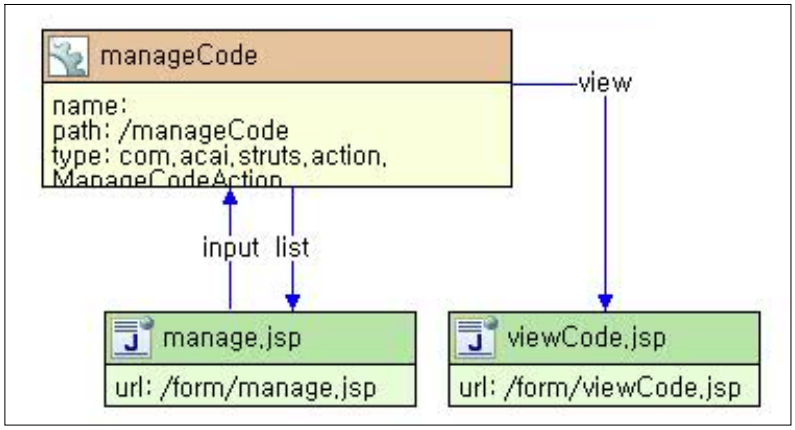

그림 3.4.15 코드 관리

다. 주요 클래스 및 메소드

1) ProductDAO

접수의 등록 및 삭제 농산물의 등록 및 삭제 등 접수와 관련된 Business Logic을 구현한 클래스로 ProductManager를 통해 접근을 한다.

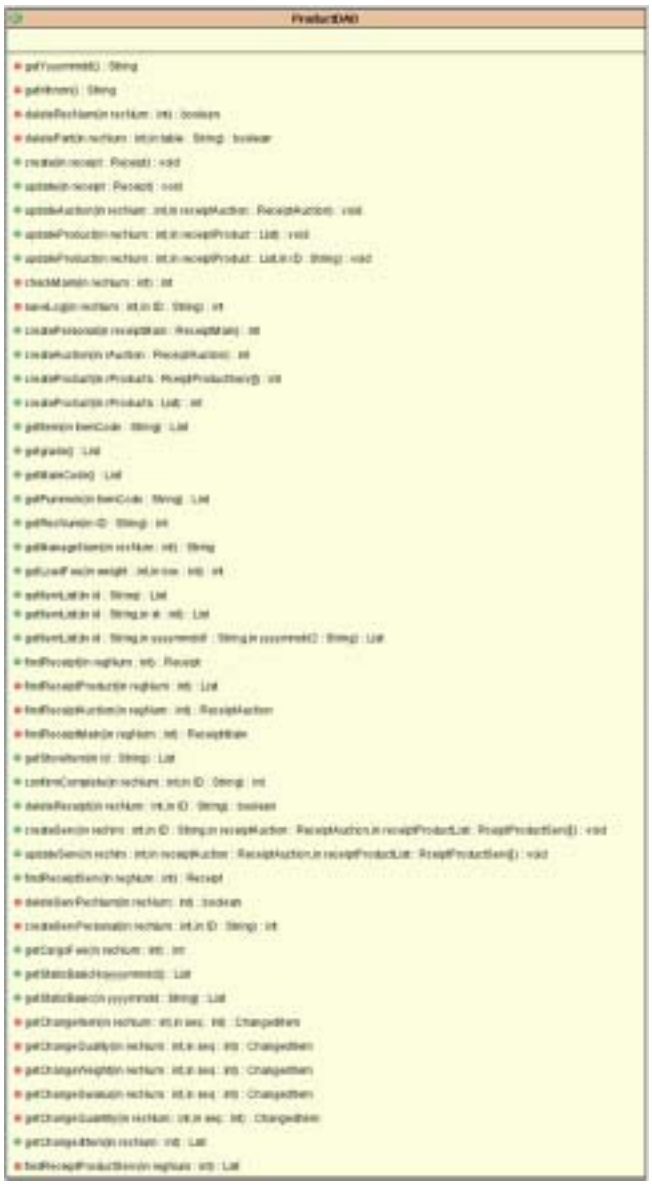

그림 3.4.16 ProductDAO 클래스 구성

- private String getYyyymmdd() : 현재 날짜를 yyyymmdd형의 문자열로 반환해 준다.
- private String getHhmm() : 현재 시간을 hhmmdd형의 문자열로 반환해 준다.
- private boolean deleteRecNum(int recNum)throws SQLException : 접수 번호가 recNum인 접수를 삭제한다.

- public void create(Receipt receipt) throws SQLException : 접수증을 생성한다.
- public void updateAuction(int recNum, ReceiptAuction receiptAuction) throws SQLException : 접수번호가 recNum인 접수증의 기타정보를 update한다.
- public void updateProduct(int recNum, List receiptProduct) throws SQLException : 출하자에 의해 접수번호가 recNum인 농산물을 변경한다.
- public void updateProduct(int recNum,List receiptProduct,String ID)throws SQLException : 접수자에 의해 접수번호가 recNum인 농산물을 변경한다. ID는 접수자의 ID를 나타낸다.
- private int checkMain(int recNum)throws SQLException : Acai_rec_num table에 특정 접수번호의 농산물이 변경 처리되었음을 알린다.
- private int saveLog(int recNum,String ID)throws SQLException : 접수자에 의한 농산물 변경 Log를 저장한다.
- public int createPersonal(ReceiptMain receiptMain)throws SQLException : 접수증의 개인정보를 추가한다.
- public int createAuction(ReceiptAuction rAuction)throws SQLException : 접수증의 기타 정보를 추가한다.
- public int createProduct(RceiptProductServ[] rProducts)throws SQLException : 접수증의 농산물정보를 추가한다. 본 메소드는 SOAP을 통한 사용자 요청에 사용된다.
- public int createProduct(List rProducts)throws SQLException : 접수증의 농산물정보를 추가한다. Internet Browser 사용자 요청에 사용된다.
- public List getItem(String itemCode) throws SQLException 품목이 itemCode인 품종을 반환한다.
- public List getgrade() throws SQLException: 등급 코드를 반환한다.
- public List getMainCode() throws SQLException : 농산물의 종류를 반환한다.
- public List getPummok(String itemCode) throws SQLException : 종류가 itemCode인 품목 코드를 반환한다.
- public int getRecNum(String ID) throws SQLException : 출하자가 접수 요청 시도가 있을 경우 새로 접수 번호를 생성하고 그 값을 반환한다, ID는 출하자 ID 이다.

- public String getManageNum(int recNum) throws SQLException : 접수가 완료된 경우 접수 번호 recNum에 대한 관리번호를 생성하고 그 값을 반환한다.
- public int getLoadFee(int weight, int box) throws SQLException : 접수가 완료된 경우 하역비를 계산하고 그 값을 반환한다.
- public List getItemList(String id) throws SQLException : 사용자 id가 id인 전 농산물을 불러온다.
- public List getItemList(String id, int st) throws SQLException : 상태가 st이며 사용자 id가 id인 농산물의 목록을 불러온다.
- public List getItemList(String id, String yyyymmdd1, String yyyymmdd2) throws SQLException : 제품의 사용자 id가 id인 제품중 등록일이 yyyymmdd1에서 yyyymmdd2사이의 제품을 불러온다.
- public Receipt findReceipt(int regNum) throws SQLException : 접수 번호가 recNum인 접수증을 반환한다.
- private List findReceiptProduct(int regNum) throws SQLException : 접수 번호가 regNum인 접수 농산물을 반환한다.
- private ReceiptAuction findReceiptAuction(int regNum) throws SQLException : 접수 번호가 recNum인 접수의 기타정보를 반환한다.
- private ReceiptMain findReceiptMain(int regNum) throws SQLException : 접수 번호가 regNum인 접수의 사용자 정보를 반환한다.
- public int confirmComplete(int recNum, String ID)throws SQLException : 접수 번호가 recNum인 접수를 ID가 ID인 접수자에 의해 입하 확인한다.
- public boolean deleteReceipt(int recNum, String ID)throws SQLException : 접수 번호가 recNum인 접수증을 ID가 ID인 관리자에 의해 삭제한다.
- public void createServ(int recNm, String ID, ReceiptAuction receiptAuction, RceiptProductServ [] receiptProductList) throws SQLException : SOAP를 통한 사용자에 의해 신규 접수증을 생성한다.
- public Receipt findReceiptServ(int regNum) throws SQLException : SOAP을 통한 사용자에 의해 접수 번호가 regNum인 접수증을 반환한다.
- private int createServPersonal(int recNum, String ID)throws SQLException : SOAP을 통한 사용자에 의해 접수증의 개인 정보를 생성한다.

- public int getCargoFee(int recNum)throws SQLException : 접수 번호가 recNum인 접수증의 하역비를 계산한다.
- public List getStaticBasicNoyyyymmdd()throws SQLException : 금일의 통계정보를 반환한다.
- public List getStaticBasic(String yyyymmdd)throws SQLException : 특정일의 통계정보를 반환한다.
- private ChangedItem getChangeItem(int recNum, int seq)throws SQLException : 접수 번호가 recNum인 접수에 대해 농산물의 변경사항을 반환한다.
- private ChangedItem getChangeQuality(int recNum,int seq)throws SQLException : 접수 번호가 recNum인 접수에 대해 등급의 변경사항을 반환한다.
- private ChangedItem getChangeWeight(int recNum,int seq)throws SQLException : 접수 번호가 recNum인 접수에 대해 무게의 변경사항을 반환한다.
- private ChangedItem getChangeGwasu(int recNum,int seq)throws SQLException : 접수 번호가 recNum인 접수에 대해 과수의 변경사항을 반환한다.
- private ChangedItem getChangeQuantity(int recNum,int seq)throws SQLException : 접수 번호가 recNum인 접수에 대해 수량의 변경사항을 반환한다.
- private List findReceiptProductServ(int regNum) throws SQLException : SOAP를 통한 사용자에 대해 접수 번호가 regNum인 농산물을 반환한다.

2) ProductManager

ProductDAO을 시행하기 위해 데이터베이스와 접근하는 클래스이다.

그림 3.4.17 ProductManager 클래스 구성

3) UtilDAO

코드 관리와 관련된 Business Logic을 구현한 클래스로 UtilManager를 통해 접근한다.

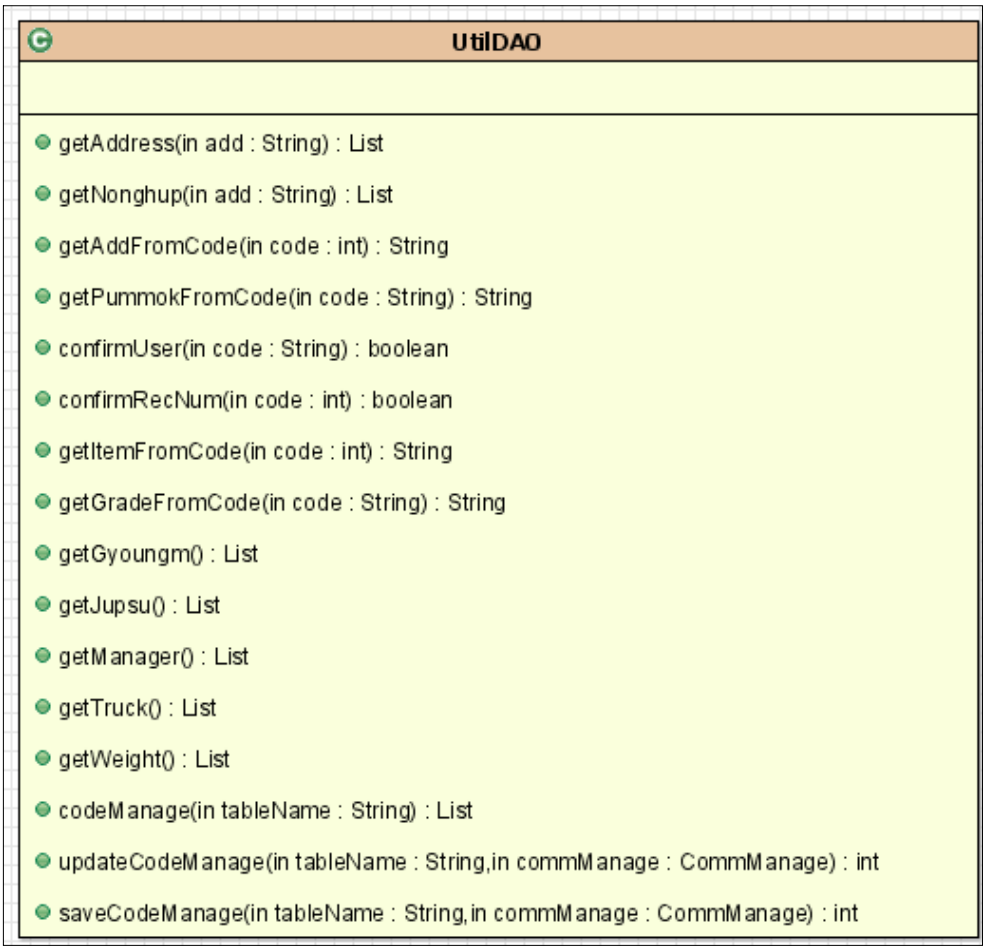

그림 3.4.18 UtilDAO 클래스 구성

- public List getAddress(String add) throws SQLException
- public List getNonghup(String add) throws SQLException
- public String getAddFromCode(int code) throws SQLException
- public String getPummokFromCode(String code) throws SQLException
- public boolean confirmUser(String code) throws SQLException
- public boolean confirmRecNum(int code) throws SQLException
- public String getItemFromCode(int code) throws SQLException
- public String getGradeFromCode(String code) throws SQLException
- public List getGyoungm() throws SQLException
- public List getJupsu() throws SQLException

- public List getManager() throws SQLException
- public List getTruck() throws SQLException
- public List getWeight() throws SQLException
- public List codeManage(String tableName) throws SQLException
- public int updateCodeManage(String tableName, CommManage commManage) throws SQLException
- public int saveCodeManage(String tableName, CommManage commManage) throws SQLException

4) UtilManager

UtilDAO을 시행하기 위해 데이터베이스와 접근하는 클래스이다.

그림 3.4.19 UtilManager 클래스 구성

5) UserDAO

사용자 관리에 관련된 Business Logic을 구현한 클래스로 UserManager를 통해 접근한다.

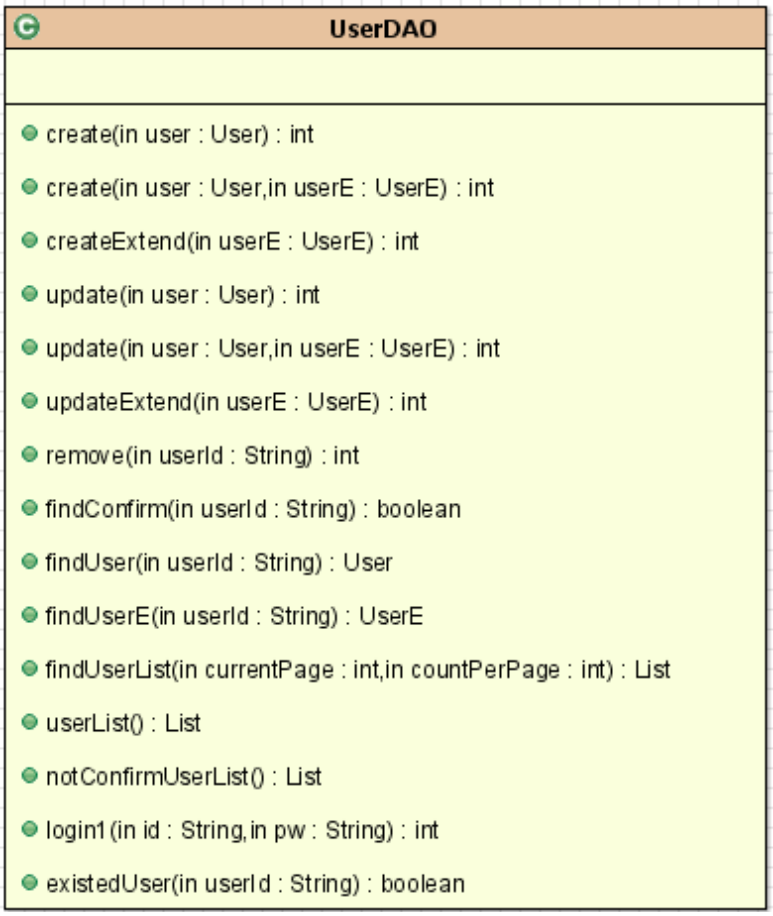

그림 3.4.20 UserDAO 클래스 구성

- public int create(User user) throws SQLException
- public int create(User user,UserE userE) throws SQLException
- public int createExtend(UserE userE) throws SQLException
- public int update(User user) throws SQLException : 기존의 사용자 사용자 정보를 수정한다.
- public int update(User user,UserE userE) throws SQLException
- public int updateExtend(UserE userE) throws SQLException
- public int remove(String userId) throws SQLException : 사용자 아이디에 해당하는 사용자를 삭제한다.
- public boolean findConfirm(String userId) throws SQLException :

- public User findUser(String userId) throws SQLException
- public UserE findUserE(String userId) throws SQLException
- public List userList() throws SQLException : 등록된 사용자를 반환한다.

- public List notConfirmUserList() throws SQLException : 등록 요청중인 사용자를 반환한다.
- public int login1(String id,String pw)throws SQLException : 시스템에 Login 한다.
- public boolean existedUser(String userId) throws SQLException : userId를 가지는 사용자가 존재하는지의 유무를 판단한다.

6) UserManager

UserDAO을 시행하기 위해 데이터베이스와 접근하는 클래스이다.

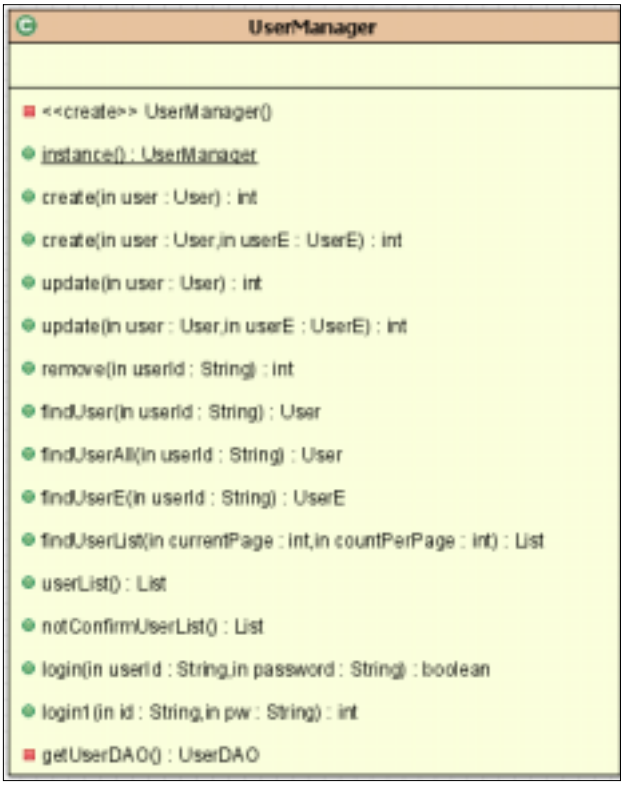

그림 3.4.21 UserManager 클래스 구성

7) UserE

Login한 사용자의 추가 정보를 저장하는 클래스

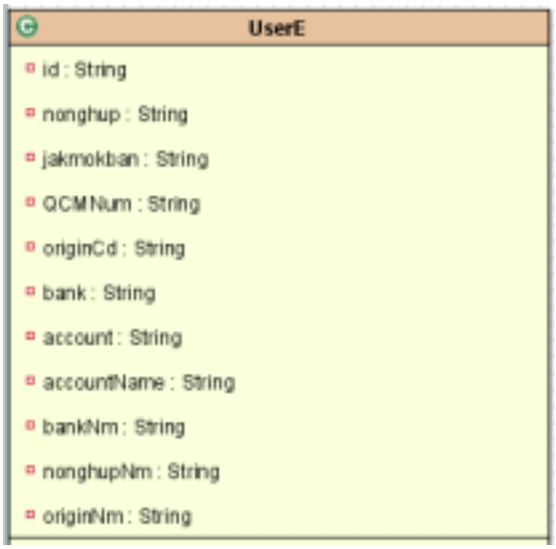

그림 3.4.22 UserE 클래스 구성

8) User

Login한 사용자의 기본정보를 저장하는 클래스로 setter, getter메소드를 갖는다.

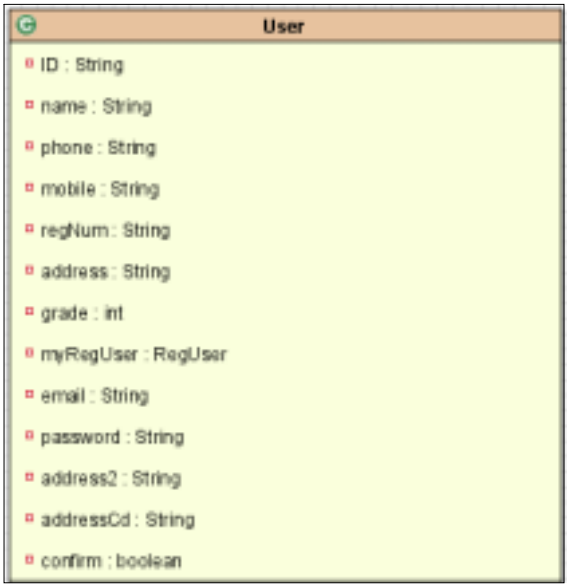

그림 3.4.23 User 클래스 구성

9) ReceiptProduct

등록 농산물에 대한 정보를 저장하는 클래스로 setter, getter메소드를 갖는다.

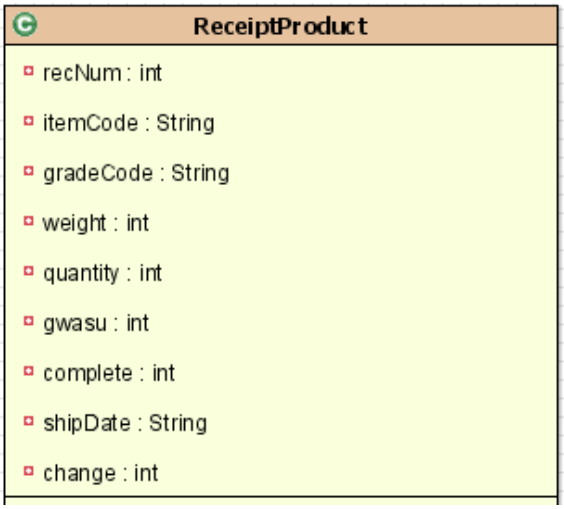

그림 3.4.24 ReceiptProduct 클래스 구성

10) ReceiptMain

접수증의 사용자 정보를 저장하는 클래스로 setter, getter메소드를 갖는다.

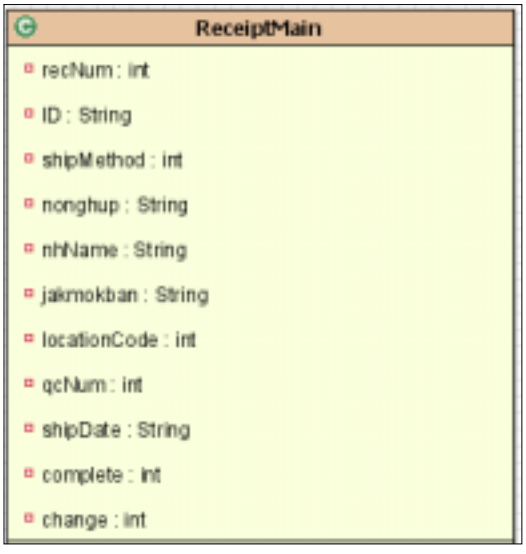

그림 3.4.25 ReceiptMain 클래스 구성

11) ReceiptAuction

접수증의 추가정보를 저장하는 클래스로 setter, getter메소드를 갖는다.

그림 3.4.26 ReceiptAuction 클래스 구성

12) Receipt

접수증에 대한 클래스이다.

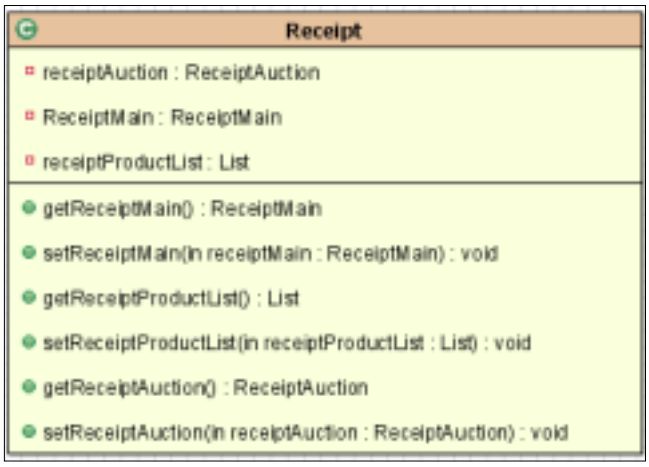

그림 3.4.27 Receipt 클래스 구성

라. 사용자 화면 설계 및 디자인

본 연구에서는 사용자를 일반 사용자, 출하자, 접수자, 관리자 등 4 등급으로 구분 하였으며 각 등급 사용자 마다 권한에 맞는 메뉴만을 사용할 수 있도록 하였다.

디자인 면에서는 편안한 색을 선택하였으며 css를 통하여 HTML단에서 디자인 요소와 데이터 요소를 구분 관리 하여 전체적인 디자인의 통일성을 유지하도록 하였다. 화면은 전체적으로 3 부분으로 구분되어지는데 상단의 로고 부분과 하단 왼쪽의 메뉴 부분 그리고 주 화면으로 구성되어져 있다. 메뉴 부분은 사용자의 권한에 맞게 재구성 되어지며 첫 화면에서 '로그인 하기', '사용자 등록 요청', '관련 홈페이지', '게시판'등의 메뉴로 구성되어져 있다.

1) 일반 사용자

일반 사용자는 아직 정식 사용이 허가되지 않은 사용자로 시스템 사용을 위한 등록 요청을 할 수 있으며 게시판을 사용할 수 있다. 일반 사용자의 메뉴는 다음과 같다.

가) 로그인 하기

나) 관련홈페이지

다) 게시판

2) 출하자

출하자는 정식 등록된 사용자로 농산물을 출하하는 농민이나 농협 담당 직원을 의미하며 접수증을 작성하는 주체이다. 출하자의 메뉴는 다음과 같다.

가) 등록상품리스트

나) 상품등록하기

다) 출하자정보수정

라) 관련홈페이지

마) 게시판

바) 현황보기

사) 로그아웃

3) 접수자

접수자는 정식 등록된 사용자로 출하자에 의해 작성되어진 접수증을 실제 입하되어진 농산물과 비교하여 입하 확인하는 역할을 담당한다. 접수자의 메뉴는 다음과 같다.

가) 접수자정보수정

나) 입하확인

다) 관련홈페이지

라) 게시판

마) 현황보기

바) 로그아웃

4) 관리자

관리자는 시스템의 전반적인 관리를 담당하며 접수증의 내용에는 관여할 수 없으며 접수증의 삭제는 가능하다. 관리자의 메뉴는 다음과 같다.

가) 사용자리스트

나) 등록요청사용자리스트

다) 접수삭제

라) 코드관리

마) 관리자정보 수정

바) 관련홈페이지

사) 게시판

아) 현황보기

자) 로그아웃

5) CSS (cascading style sheet)

CSS는 웹페이지용 스타일 시트로서, 상충되는 스타일 요소의 정의가 있는 곳마다 서열을 명확하게 한 다수의 근거로부터 파생되었다. W3C의 권고안으로서 넷스케이프와 마이크로소프트 웹브라우저에 구현되어 있는 CSS1은, 주어진 요소가 웹페이지 내에 어떻게 표현되는지를 결정할 수 있는 가능한 스타일 시트들과 문장들을 정의한다. CSS는 브라우저 설계자나 독자에게보다는 페이지 제작자에게, 웹페

이지 외관을 통제할 수 있는 더 많은 권한을 준다. 주어진 문서 구성요소에 대한 스타일 정의의 근거인 CSS에는, 다음과 같은 서열이 있다.

가) 개별 구성요소 태그 상에 있는 STYLE 속성

나) 스타일 선언을 포함하는 특정 스타일 시트를 정의하는 STYLE 구성요소, 또는 STYLE 구성다)요소를 포함하는 별도의 문서를 링크하는 LINK 요소. 웹 페이지에서, STYLE 구성요소는 TITLE문과 BODY문 사이에 위치한다.

라) 불러들인 스타일 시트, 즉 자동으로 외부 스타일 시트를 불러들여 현재의 스타일 시트에 병합하기 위해 CSS @import 표기를 사용한다.

마) 독자가 브라우저에 정의한 스타일 속성들

바) 브라우저에 의해 상정된 기본 스타일 시트

본 연구에서는 CSS를 통해 디자인과 데이터를 분리 관리하였다. 본 연구에서 사용되어진 CSS는 아래와 같다.

```
/* Basic */
td {font-family:Dotum; font-size:12px; color:#787878; line-height:16px}
a, a:active, a:focus{font-family:Dotum; font-size:12px; text-decoration:none; color:#787878;}
a:link,  a:visited {font-family:Dotum; font-size:12px; text-decoration:none; color:#1A80CF;}
a:hover {font-family:Dotum; font-size:12px; text-decoration:underline; color:#18BFE0;}
img {border:0;}

/*INPUT*/
input, select {font-family:Dotum; font-size:12px; color:#545454;background-color: #ffffff; }
.radio_input{font-family:Dotum; font-size:12px; color:#545454; background-color: #ffffff }
```

```
.button_input{font-family:Dotum; font-size:12px; color:#B30707;border: 1px #cccccc solid; background-color:#e7e7e7}
.samewidth_input{font-family:Dotum; font-size:12px; color:#B30707;border: 1px #cccccc solid; background-color:#e7e7e7; width:100;}

/*01_about*/
.01_small {font-family:Dotum; font-size:11px; color:#747474; line-height:16px}
.01_small_color01 {font-family:Dotum; font-size:11px; color:#ff6600; line-height:16px}
.01_small_color02 {font-family:Dotum; font-size:11px; color:#729B0F; line-height:16px}
.01_small_cate {font-family:Dotum; font-size:11px; color:#279FF6; line-height:16px}
.01_light_color {font-family:Dotum; font-size:12px; color:#0763BA; line-height:16px}
.01_bold_color {font-family:Dotum; font-size:12px; font-weight:bold; color:#0763BA; line-height:16px}
.style1 {font-family:Dotum; font-size:12px; color:#ffffff; line-height:16px}

/*quick*/
.quick_small {font-family:Dotum; font-size:11px; color:#330000; line-height:16px}
.quick_name {font-family:Dotum; font-size:12px; color:#18878E; line-height:16px}

A.quick_LINK:link{font-family:Dotum; font-size:11px; color:330000; text-decoration:none}
A.quick_LINK:visited{font-family:Dotum; font-size:11px; color:330000; text-decoration:none}
```

```
A.quick_LINK:hover{font-family:Dotum; font-size:11px; color:006666; text-decoration:underline}

/* Class */
.searchbox_board {font-family:Dotum; font-size:12px; border:1 solid #D5D4CE; color:#666666; background:#ffffff; height:18;}
.searchbox_text {font-family:Dotum; font-size:12px; border:1 solid #666666; color:#666666; background:#ffffff; height:18;}
.searchbox_all {font-family:Dotum; font-size:12px; border:1 solid #CBC3A2; color:#666666; background:#ffffff; height:18;}

/* Link */
A.BOARD01:link{font-family:Dotum; color:747474; text-decoration:none}
A.BOARD01:visited{font-family:Dotum; color:747474; text-decoration:none}
A.BOARD01:hover{font-family:Dotum; color:9A7F04; text-decoration:underline}

A.g_BOARD01:link{font-family:Dotum; color:747474; text-decoration:none}
A.g_BOARD01:visited{font-family:Dotum; color:747474; text-decoration:none}
A.g_BOARD01:hover{font-family:Dotum; color:5095A8; text-decoration:underline}

/* board_sub01 */
.board_title {font-family:Dotum; font-size:12px; color:#1E8533; font-weight:bold; background:#F2FBF4; height:26px;}
.board_title_f7f7f7 {background:#f7f7f7; height:30px; font-family:Dotum; font-size:12px; font-weight:bold; color:#0763BA;}
.board_line1px_cccccc {background:#cccccc; height:1px;}
.board_line1px_e4e4e4 {background:#e4e4e4; height:1px;}
```

```
.board_line2px_cccccc {background:#cccccc; height:2px;}
.board_line2px_e4e4e4 {background:#e4e4e4; height:2px;}
.board_no {font-family:Dotum; font-size:11px; color:#747474; background:#FFFFFF; height:26px;}
.board_title02 {font-family:Dotum; font-size:12px; color:#747474; background:#FFFFFF; height:26px;}
.nomal_line {background:#ECECEC;}
.board_search_bg {background:#F9F9F9; height:20px;}
.down_search_bg {background:#F9F9F9; height:20px;}
.board_number {font-family:Dotum; font-size:11px; color:#5B5B5B; padding:0 0 0 0;}
.board_number_b {font-family:Dotum; font-size:11px; font-weight:bold; color:#279FF6;}

/* text_01 */
.text_board_title {font-family:Dotum; font-size:11px; color:#7B7B7B; font-weight:bold; padding:4 0 0 10; background:#FFFFFF; height:26px;}
.text_board_title_p {font-family:Dotum; font-size:11px; color:#FF6600; font-weight:bold; padding:4 0 0 10; background:#FFFFFF; height:26px;}
.text_board_title02 {font-family:Dotum; font-size:11px; color:#7B7B7B; padding:0 0 0 10; background:#FFFFFF; height:26px;}
.text_board_title03 {font-family:Dotum; font-size:11px; color:#7B7B7B; padding:4 0 0 0; background:#FFFFFF;}
.text_board_title_sbg {background:#ffffff; height:10px;}
.text_board_line3px {background:#5095A8; height:3px;}
.text_board_line2px {background:#5095A8; height:2px;}
.text_board_space2px {background:#FFFFFF; height:5px;}
.text_board_no {font-family:Dotum; font-size:11px; color:#747474; background:#FFFFFF; padding:0 5 0 5; height:26px;}
.text_board_event {font-family:Dotum; font-size:11px; color:#ff6600;}
```

```
.text_board_title {font-family:Dotum; font-size:12px; color:#747474; backgro
und:#FFFFFF; height:26px;}
.text_nomal_line {background:#ECECEC;}
.text_board_search_bg {background:#F9F9F9; height:26px;}
.text_down_search_bg {background:#F9F9F9; height:29px;}
```

마. WebService를 위한 원격 객체 정의 및 구현

본 연구에서는 Tomcat 기반에 Axis를 이용하여 Webservice를 구현하였다.

1) Axis

Axis는 2000년 말에 IBM의 "SOAP4J"에서 시작하는 Apache SOAP의 세 번째 생성물이다. Apache SOAP v2의 커밋터(committers)는 엔진의 좀더 유연하고, 설정가능하고, W3C의 XML프로토콜과 SOAP를 모두 다룰 수 있다.

가) Axis는 다음의 핵심기능

- 속도: Axis는 Apache SOAP의 기존 버전보다 좀더 빠른 속도를 달성할 수 있는 SAX(이벤트기반의) 파서를 사용한다.
- 유연성: Axis구조는 변경된 헤더처리, 시스템관리 또는 당신이 가정할 수 있는 다른 것을 위해 엔진으로 확장된 것을 추가할 수 있는 완벽한 자유를 준다.
- 안정: Axis는 Axis의 나머지에 비교적으로 천천히 비교되는 변경 점을 가지는 published interfaces의 세트를 정의한다.
- 컴포넌트 기반 배치: 자신의 애플리케이션이나 파트너에게 배포하기 위한 프로세스의 공통적인 패턴을 구현하기 위한 핸들러(Handlers)의 재사용가능한 네트워크를 정의한다.
- Framework 옮기기: 트랜스포트(transports - 예를 들면 SMTP, FTP, 메시지 기반 미들웨어와 같은 다양한 프로토콜위에서 SOAP를 위한 샌더(sender)와 리스너(listener))을 디자인하기 위해 깔끔하고 간단한 추상화를 가진다. 그리고 엔진의 핵심은 완벽하게 트랜스포트(transport)에 비의존적이다.

- WSDL 지원: Axis는 웹서비스 서술언어(Web Service Description Language) 1.1을 지원한다. 원격서비스에 접근하기 위한 스텁(stub)를 쉽게 빌드하도록 하고 Axis로부터 배치된 서비스의 기계가 읽을 수 있는 서술을 자동적으로 보낼 수 있다.

나) Axis 구조

Axis는 가장 유연하고 멋진 아키텍쳐를 가진 웹서비스 엔진이다. 다음 그림은 헨들러와 체인의 구조이다.

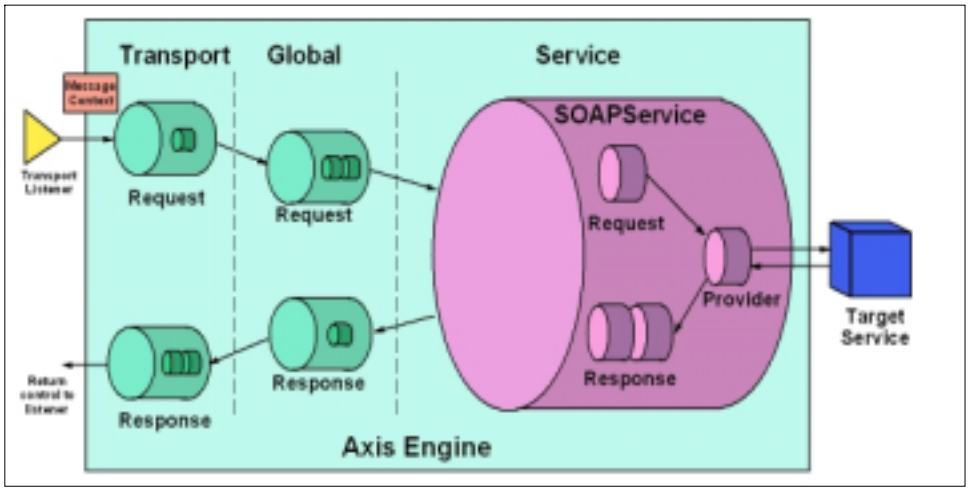

그림 3.4.28 AXIS 의 서버측 구조

이전의 초기 웹서비스 엔진들은 SOAP 메세지가 도착하면 이것을 받아 단순히 backend 객체를 연결해 주는 역할을 하였다. 따라서 객체가 호출되기 전에 하였으면 하는 전처리가 쉽지 않았던 것이다. 하지만 Axis에서는 헨들러를 두어 이것을 해결하고 있다. 메세지가 도착하면 Transport => Global => Service 헨들러 순으로 호출이 되어지는 것이다. 그리고 각각의 헨들러 에는 Chain을 두어 동일 계층에서 수행할 수 있는 복수개의 헨들러를 정의할 수 있다. 헨들러 간에는 SOAP 메세지가 MessageContext 라고 하는 고유의 객체로 변형되어 전달되어 지는데 이 안에는 ‘ SOAP 원본 메세지 + 부가 내용’ 들이 포함되게 된다.

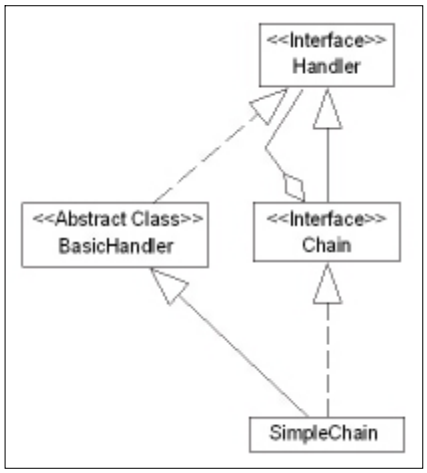

그림 3.4.29 Chain의 구조

Chain은 자체가 헨들러를 상속 받고 또한 헨들러를 Aggregation 하고 있다. 그리고 앞에서 언급한 MessageContext 는 아래와 같은 구조를 가진다.

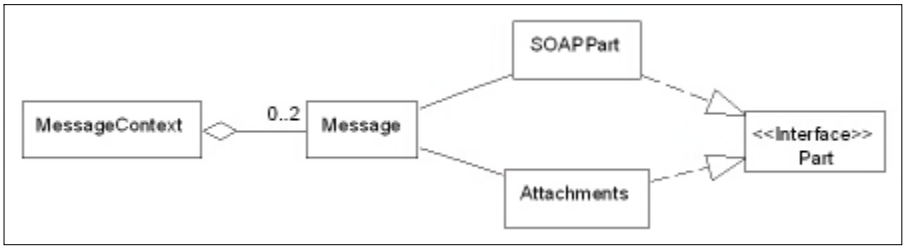

그림 3.4.30 MessageContext 구조

SOAP 표준에 정해 진바와 같이 SOAP 메시지와 Attachment로 이루어 져 있는 것을 볼 수 있다. 그리고 Message 클래스의 multiplicity 가 0..2인 이유는 Request 와 Response 모두를 가지게 되기 때문이다. 다음 그림은 SOAP 메세지의 표준 전체 모습이다.

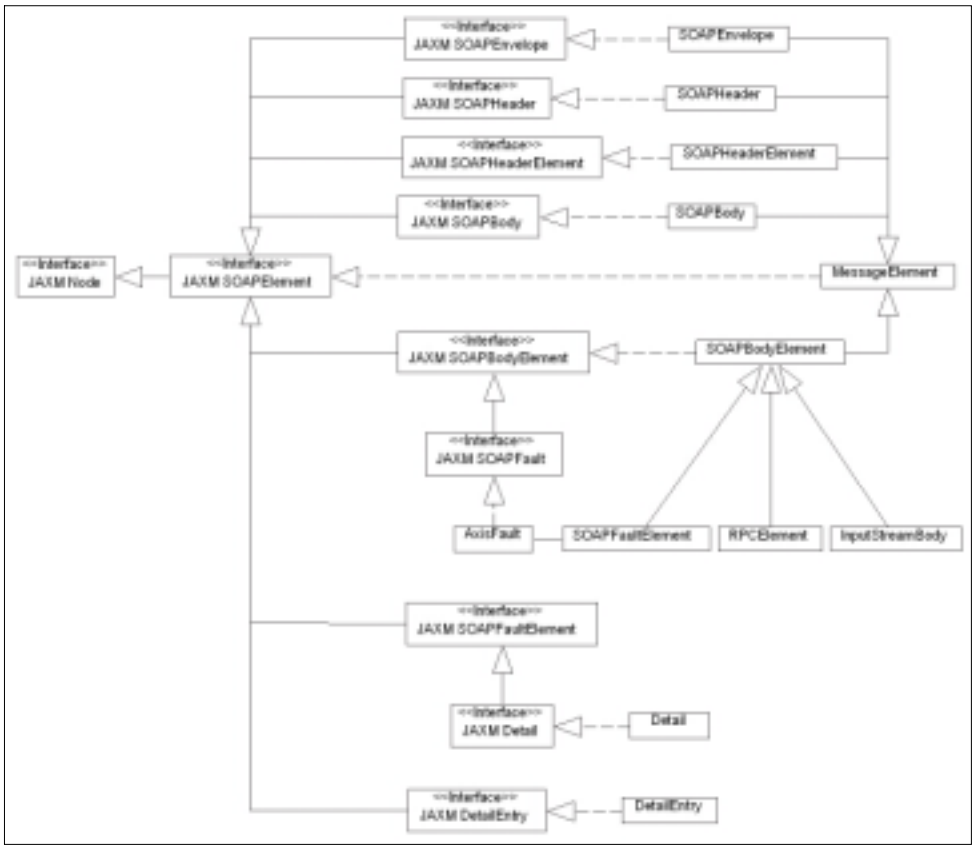

그림 3.4.31 SOAP 메시지의 구조

JAXM은 일종의 wrapper라고 할 수 있다.. 즉, SOAP 메세지를 생성하여 JAXM 형식으로 즉, Document 형식으로 전송할 때에 Wrapper가 사용되는 것이기 때문에 코딩 시에는 전혀 고려하지 않아도 된다. 단, 이를 처리하는 웹서비스 엔진에서는 이것이 Document 방식의 SOAP 메세지라는 것을 알아야 하는 것이다. 클라이언트가 웹서비스를 호출하는 과정은 다음과 같다.

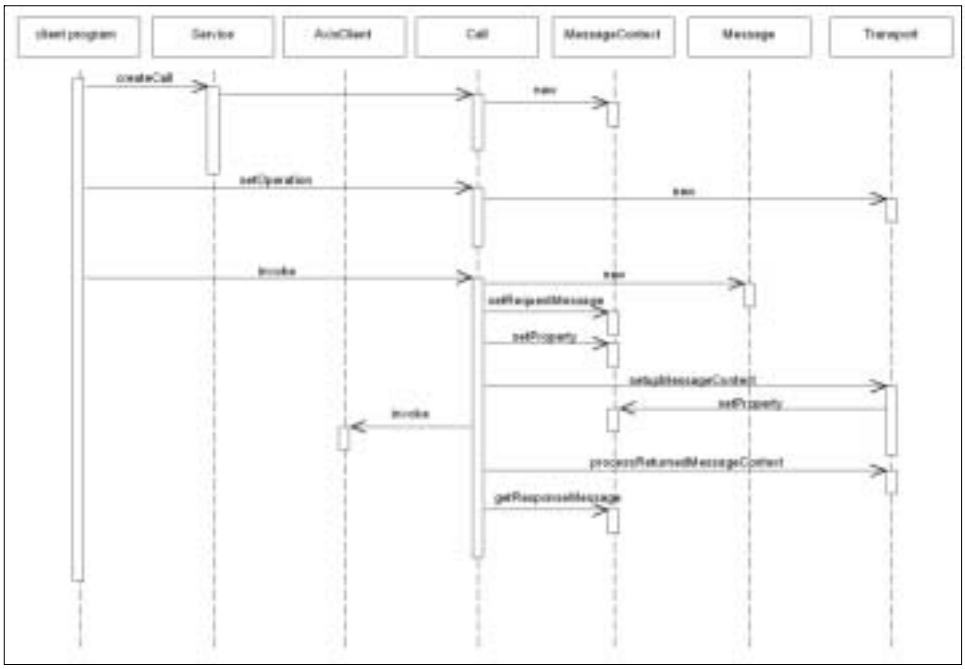

그림 3.4.32 웹서비스 호출 과정

2) 외부 서비스를 위한 WSDL

SOAP를 통한 원격 객체 참조가 이루어지게 하기위해서는 표준화된 통신 규약에 대한 기술서가 필요한데 WSDL은 SOAP을 통한 객체 통신에 대한 통신 규약을 기술한 기술서이다. 따라서 서버에서는 우선 WSDL을 공개하여 서버의 서비스를 사용하고자 하는 클라이언트 및 다른 서버에게 사용방법을 기술하여 준다. WSDL의 내용에 대한 기술은 WDDI에 의해 기술되어지나 본 연구에서는 단일 목적으로 SOAP를 사용하여 불특정 다수에 대한 서비스 개방이 아닌 특정 대상에 대한 서비스 개방을 시행함으로 별도의 WDDI 서비스는 운영하지 않고 직접 WSDL을 통한 접근 인터페이스를 구성하도록 하였다. 또한 본 시스템의 PDA Application Program은 접수자용 프로그램과 출하자용 프로그램으로 구성되어져 있으나 서비스는 동일한 WSDL을 통하여 이루어지도록 하였다.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions targetNamespace="http://userAuth.webservice" xmlns:a
```

pachesoap="http://xml.apache.org/xml-soap" xmlns:impl="http://userAuth.webservice" xmlns:intf="http://userAuth.webservice" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns1="http://unit.acai.com" xmlns:tns2="http://main.acai.com" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
- <!--
WSDL created by Apache Axis version: 1.4
Built on Dec 09, 2005 (03:14:07 GMT+00:00)

  -->
- <wsdl:types>
- <schema targetNamespace="http://main.acai.com" xmlns="http://www.w3.org/2001/XMLSchema">
  <import namespace="http://userAuth.webservice" />
  <import namespace="http://unit.acai.com" />
  <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
- <complexType name="RegUser">
- <sequence>
  <element name="user" nillable="true" type="tns1:User" />
  <element name="myUser" nillable="true" type="tns1:User" />
  </sequence>
  </complexType>
  </schema>
- <schema targetNamespace="http://unit.acai.com" xmlns="http://www.w3.org/2001/XMLSchema">
  <import namespace="http://userAuth.webservice" />
  <import namespace="http://main.acai.com" />
  <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
- <complexType name="User">

```
- <sequence>
  <element name="ID" nillable="true" type="xsd:string" />
  <element name="address" nillable="true" type="xsd:string" />
  <element name="address2" nillable="true" type="xsd:string" />
  <element name="addressCd" nillable="true" type="xsd:string" />
  <element name="confirm" type="xsd:boolean" />
  <element name="email" nillable="true" type="xsd:string" />
  <element name="grade" type="xsd:int" />
  <element name="mobile" nillable="true" type="xsd:string" />
  <element name="myRegUser" nillable="true" type="tns2:RegUser" />
  <element name="name" nillable="true" type="xsd:string" />
  <element name="password" nillable="true" type="xsd:string" />
  <element name="phone" nillable="true" type="xsd:string" />
  <element name="regNum" nillable="true" type="xsd:string" />
  </sequence>
  </complexType>
- <complexType name="UserE">
- <sequence>
  <element name="QCMNum" nillable="true" type="xsd:string" />
  <element name="account" nillable="true" type="xsd:string" />
  <element name="accountName" nillable="true" type="xsd:string" />
  <element name="bank" nillable="true" type="xsd:string" />
  <element name="bankNm" nillable="true" type="xsd:string" />
  <element name="id" nillable="true" type="xsd:string" />
  <element name="jakmokban" nillable="true" type="xsd:string" />
  <element name="nonghup" nillable="true" type="xsd:string" />
  <element name="nonghupNm" nillable="true" type="xsd:string" />
  <element name="originCd" nillable="true" type="xsd:string" />
  <element name="originNm" nillable="true" type="xsd:string" />
  </sequence>
```

```
  </complexType>
- <complexType name="ReceiptProduct">
- <sequence>
  <element name="change" type="xsd:int" />
  <element name="complete" type="xsd:int" />
  <element name="gradeCode" nillable="true" type="xsd:string" />
  <element name="gwasu" type="xsd:int" />
  <element name="itemCode" nillable="true" type="xsd:string" />
  <element name="quantity" type="xsd:int" />
  <element name="recNum" type="xsd:int" />
  <element name="shipDate" nillable="true" type="xsd:string" />
  <element name="weight" type="xsd:int" />
  </sequence>
  </complexType>
- <complexType name="ReceiptAuction">
- <sequence>
  <element name="ACCOUNT_NUM" nillable="true" type="xsd:string" />
  <element name="CARGO_FEE" type="xsd:int" />
  <element name="DEALER_ID" type="xsd:int" />
  <element name="EXPENSE" type="xsd:int" />
  <element name="LOW_COST" type="xsd:int" />
  <element name="MANAGER_ID" type="xsd:int" />
  <element name="MANAGE_NUM" nillable="true" type="xsd:string" />
  <element name="RECEIPT" nillable="true" type="xsd:string" />
  <element name="REC_NUM" type="xsd:int" />
  <element name="REST" nillable="true" type="xsd:string" />
  <element name="dealerName" nillable="true" type="xsd:string" />
  <element name="managerName" nillable="true" type="xsd:string" />
  <element name="receiptName" nillable="true" type="xsd:string" />
  <element name="truckId" type="xsd:int" />
```

```
  <element name="truckNum" nillable="true" type="xsd:string" />
  </sequence>
  </complexType>
- <complexType name="ReceiptMain">
- <sequence>
  <element name="ID" nillable="true" type="xsd:string" />
  <element name="change" type="xsd:int" />
  <element name="complete" type="xsd:int" />
  <element name="jakmokban" nillable="true" type="xsd:string" />
  <element name="locationCode" type="xsd:int" />
  <element name="nhName" nillable="true" type="xsd:string" />
  <element name="nonghup" nillable="true" type="xsd:string" />
  <element name="qcNum" type="xsd:int" />
  <element name="recNum" type="xsd:int" />
  <element name="shipDate" nillable="true" type="xsd:string" />
  <element name="shipMethod" type="xsd:int" />
  </sequence>
  </complexType>
- <complexType name="RceiptProductServ">
- <sequence>
  <element name="complete" type="xsd:int" />
  <element name="grade" nillable="true" type="xsd:string" />
  <element name="gradeCode" nillable="true" type="xsd:string" />
  <element name="gwasu" type="xsd:int" />
  <element name="item" nillable="true" type="xsd:string" />
  <element name="itemCode" nillable="true" type="xsd:string" />
  <element name="pummok" nillable="true" type="xsd:string" />
  <element name="quantity" type="xsd:int" />
  <element name="recNum" type="xsd:int" />
  <element name="shipDate" nillable="true" type="xsd:string" />
```

<element name="weight" nillable="true" type="xsd:string" />
<element name="weightCode" type="xsd:int" />
</sequence>
</complexType>
- <complexType name="Receipt2">
- <sequence>
<element name="receiptAuction" nillable="true" type="tns1:ReceiptAuction" />
<element name="receiptMain" nillable="true" type="tns1:ReceiptMain" />
<element name="receiptProductList" nillable="true" type="impl:ArrayOf_tns2_RceiptProductServ" />
</sequence>
</complexType>
- <complexType name="CommManage">
- <sequence>
<element name="name" nillable="true" type="xsd:string" />
<element name="no" type="xsd:int" />
<element name="use" type="xsd:boolean" />
</sequence>
</complexType>
- <complexType name="Grade">
- <sequence>
<element name="code" nillable="true" type="xsd:string" />
<element name="name" nillable="true" type="xsd:string" />
</sequence>

이하 부록1 참조

5. Web Browser 기반의 클라이언트 시스템 구축

클라이언트 부분은 HTML로 먼저 화면 디자인을 하고 JSP로 전환하여 Business Logic과 연계하도록 구성하였다. 효과적인 MVC(Model View Control) 모델 구현을 위해 JSP로 구현되는 부분에서는 Business Logic을 최대한 배제하고 페이지 이동시에도 Control Servlet에서 경로 이동을 담당하도록 하였다.

가. WSDL을 이용한 클래스 구현

SOAP 통신을 위한 클래스를 다음과 같이 구현 하였다.

1) UserAuthIF

SOAP 통신을 위한 Interface를 정의한 클래스로 원격 객체 사용자가 사용할 수 있는 method에 대해 정의되어져 있다.

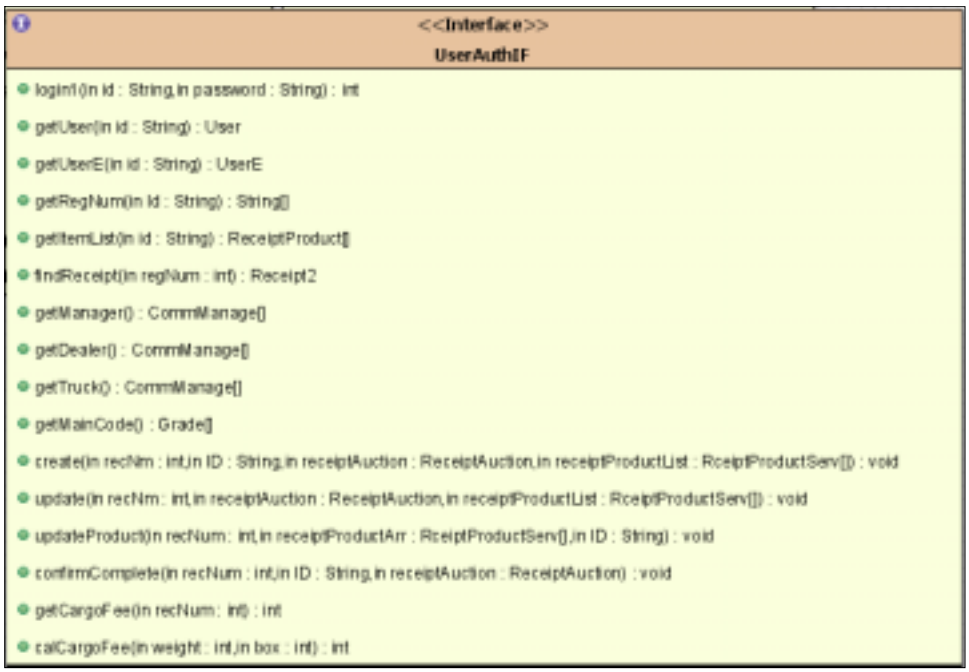

그림 3.5.1 UserAuthIF 클래스 구조

2) UserAuthImpl

UserAuthIF에서 정의되어진 method에 대한 실행 Logic을 구현한 클래스이다.

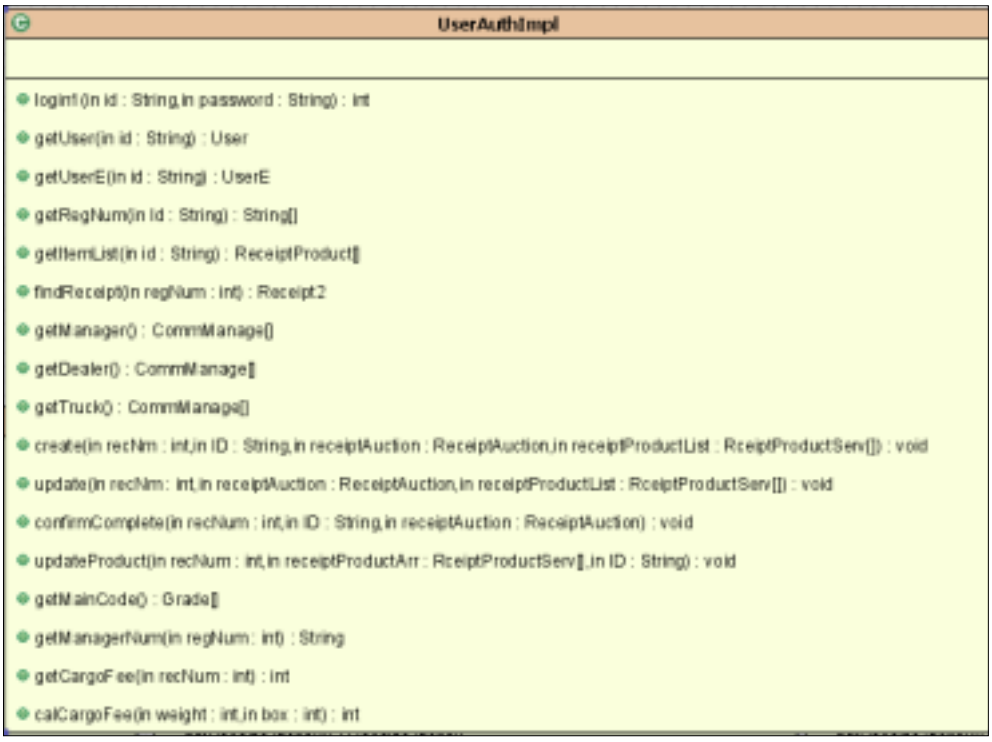

그림 3.5.2 UserAuthImpl 클래스 구조

가) public int login1(String id,String password) : 사용자 로그인을 하는 메소드이다. 입력파라미터로 사용자 id와 비밀번호를 받으며 결과 값으로 사용자의 등급 코드를 반환한다.

나) public UserE getUserE(String id): 추가 사용자 정보를 반환한다. 입력 값으로 사용자 id를 받으며 결과 값으로 사용자 추가 정보 클래스를 반환한다. public int getRegNum(String Id): 접수 번호를 반환한다. 입력 값으로 사용자 id를 받으며 결과 값으로 접수 번호를 반환한다.

다) public ReceiptProduct[] getItemList(String id, int st): 특정 사용자에 의해 등록되어진 농산물의 리스트를 반환한다. 입력 값으로 사용자 id와 농산물의 상태를 받으며 결과 값으로 농산물 정보 클래스의 배열을 반환한다.

라) public Receipt2 findReceipt(int regNum): 접수 번호에 해당하는 접수 정보를

반환한다. 입력 값으로 접수 번호를 받으며 출력 값으로 접수정보를 반환한다.

마) public CommManage[] getManager(): 접수증에서 사용되어지는 관리자 리스트를 반환한다.

바) public CommManage[] getDealer(): 접수증에서 사용되어지는 경매사 리스트를 반환한다.

사) public CommManage[] getTruck(): 접수증에서 사용되어지는 트럭 리스트를 반환한다.

아) public void create(int recNm,String ID ,ReceiptAuction receiptAuction, RceiptProductServ [] receiptProductList): 접수증 등록을 한다. 입력 값으로 접수자 id, 접수 번호, 접수 농산물 정보, 기타 정보를 받는다.

자) public void update(int recNm,ReceiptAuction receiptAuction, RceiptProductServ [] receiptProductList): 접수증의 내용을 update한다. 입력 값으로 접수 번호, 접수 농산물 정보, 기타 정보를 받는다.

차) public void confirmComplete(int recNum, String ID, ReceiptAuction receiptAuction): 입하 확인을 한다. 입력 값으로 접수 번호 사용자 id 기타 정보를 받는다.

타) public void updateProduct(int recNum, RceiptProductServ [] receiptProductArr, String ID): 농산물 정보를 update한다. 접수자에 의해 농산물 정보가 변경 되었을 경우 사용되어진다. 입력 값으로 접수 번호 접수자 id 변경 농산물 정보를 받는다.

카) public Grade[] getMainCode() : 농산물 등급 코드를 반환 받는다.

파) public String getManagerNum(int regNum): 관리 번호를 반환한다. 입력 값으로 접수 번호를 받는다.

하) public int getCargoFee(int recNum): 하역비를 반환 한다. 입력 값으로 접수 번호를 받는다.

3) Receipt2

원격 객체에서 사용하는 접수증 객체이다. Receipt2 클래스는 멤버 클래스로 출하자 정보 클래스 농산물 정보 클래스 기타관리 클래스를 갖는다. getter, setter m

ethod로 구성되어져 있다.

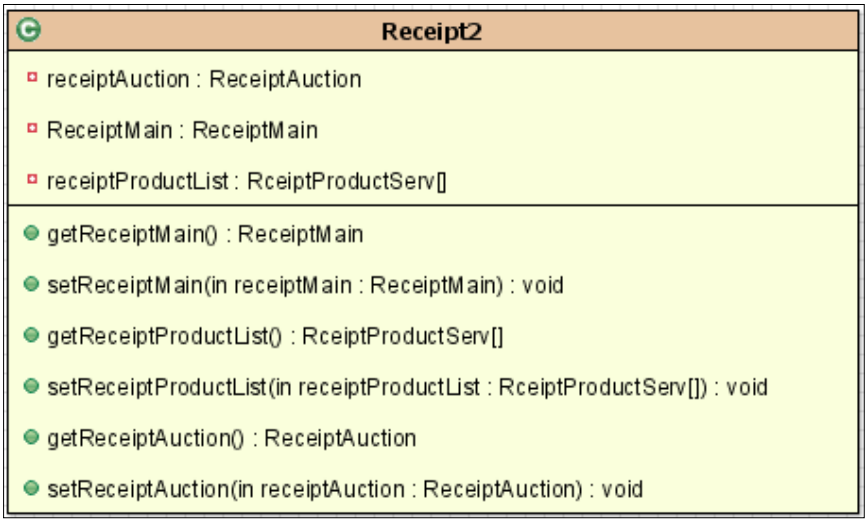

그림 3.5.3 Receipt2 클래스 구조

4) ReceiptProductServ

원격 객체 사용자가 사용하는 농산물에 대한 리스트 정보가 들어 있는 클래스이다. getter, setter method로 구성되어져 있다.

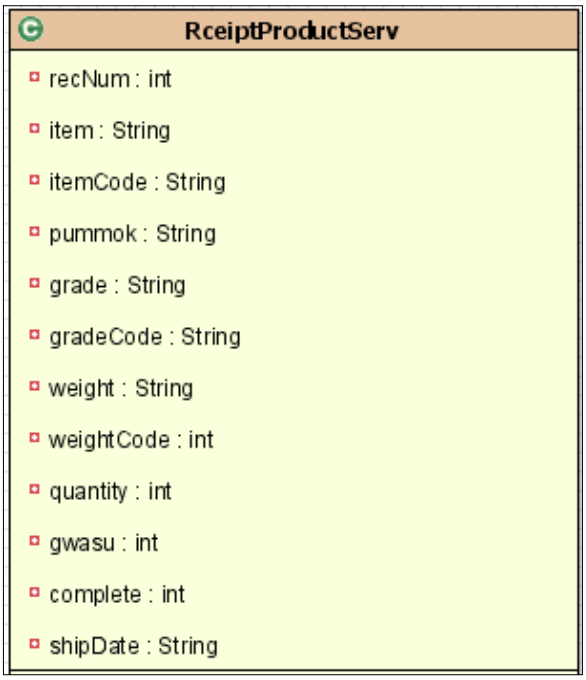

그림 3.5.4 ReceiptProductServ클래스 구조

나. 사용자 화면의 구성

디자인 되어진 HTML 파일 기본으로 하여 시스템에서 이용 가능한 JSP 파일로 변환 하였다. HTML은 정적인 페이지 구성만 가능하기 때문에 시스템의 응답에 대한 적절한 내용을 표시해 줄 수 없다. 그러나 JSP는 서버에 의해 동적으로 구성되어진 페이지로 동적 페이지 응답이 가능하다.

1) JSP

JSP는 클라이언트의 요청을 받아 동적인 웹페이지 생성을 위한 자바 플랫폼 기술이다. JSP는 자바 플랫폼 기술이므로 자바가 지원하는 API를 이용할 수 있으며 자바 기반의 여타 기술을 이용할 수 있다. JSP 페이지는 클라이언트의 요청을 받으면 자바 코드로 컴파일되고 실행된 후 그 결과가 클라이언트로 전송이 된다. 자바 코드로 컴파일되고 실행될 때는 서블릿의 인스턴스가 생성된다. 이 인스턴스는 한 번만 생성되며 이후의 요청은 이 인스턴스의 쓰레드로서 처리된다.

2) CGI와 비교

CGI와 수행 성능 면에서 많은 차이가 있다. CGI는 요청되는 개수만큼 프로세스

가 생성되므로 자원이용이 비효율적이 될 수밖에 없다. 즉 JSP는 자원이용의 효율성을 꽤할 수 있으며 자바기반의 여타 기술을 이용할 수 있다는 것이다.

3) JSP의 동작

다음 그림은 JSP가 동작하는 원리에 대한 것이다.

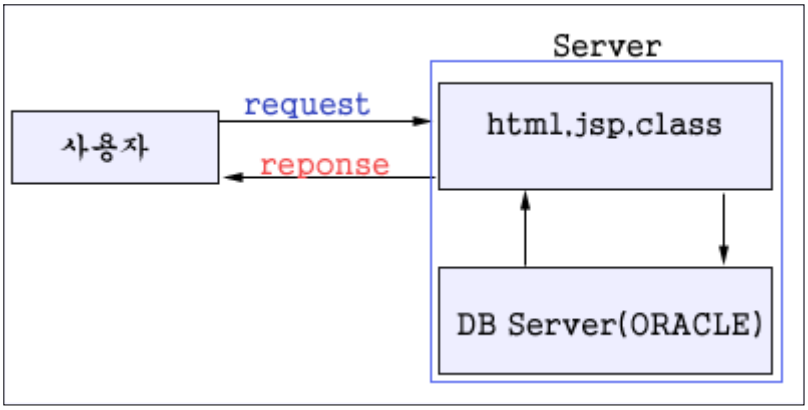

그림 3.5.5 JSP 동작

가) 클라이언트가 서버에 Page를 요구한다.

나) 웹서버는 요구받은 JSP문서을 분석한다.

다) JSP문서에서 필요한 Class를 로딩한다.

라) 해당 Class가 DB를 필요로 할 경우 DB에서 데이터를 불러온다.

마) DB에서 불러온 데이터를 JSP파일에 붙여넣는다.

바) 웹서버는 지금 까지 처리된 내용을 html형식으로 클라이언트에게 보내준다

4) 디자인 화면의 JSP로 변환

디자인이 완료되어진 화면은 정적 화면으로 서버와의 연동을 통한 운영을 하기 위해서는 동적 화면으로 변환이 필요하다. 본 연구에서는 HTML로 디자인 되어진 화면을 JAVA 기반의 JSP를 사용하여 변환하였다. JSP로 작성되어진 화면은 서버에서 JAVA 코드로 변환 되어 관리되어지며 사용자의 요청이 있어 응답을 할 경우에는 HTML 형태로 전송되어진다. java script로 작성되어진 코드는 사용자에게 보여지나 JSP에서 java로 작성되어진 코드는 보이지 않는다. 아래는 상품 등록 화

면의 변환 과정을 보여주고 있다.

가) 디자인 화면 코드

디자인 화면의 코드는 HTML로 작성하였다.

```
<td height="23" align="center" >
            <input type="submit"class="samewidth_input" style="cursor:ha
nd" name="type0" value="과실류"OnClick="goConfirm('0','06')">
        </td>
        <td height="23" align="center" >
            <input type="submit"  class="samewidth_input" style="cursor:
hand"name="type1" value="과일과채류"OnClick="goConfirm('0','08')">
        </td>
        <td height="23" align="center" >
            <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type2" value="과채류"
              OnClick="goConfirm('0','09')" >
        </td>
        <td height="23" align="center" >
            <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type3" value="근채류"
              OnClick="goConfirm('0','11')" >
        </td>
        <td height="23" align="center" >
            <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type4" value="두류"
              OnClick="goConfirm('0','03')" >
        </td>
        <td height="23" align="center" >
            <input type="submit"  class="samewidth_input" style="cursor:
```

```
hand" name="type5" value="맥류"
                OnClick="goConfirm('0','02')" >
          </td>
        </tr>
        <tr class="board_line1px_e4e4e4">
          <td height="1" colspan="6" align="center"></td>
        </tr>
      <tr>
          <td height="23" align="center" >
             <input type="submit"  class="samewidth_input" style="cursor:h
and" name="type6" value="미곡류"
                OnClick="goConfirm('0','01')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type7" value="버섯류"
                OnClick="goConfirm('0','17')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type8" value="산채류"
                OnClick="goConfirm('0','14')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type9" value="서류"
                OnClick="goConfirm('0','05')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
```

```
hand" name="type10" value="수실류"
                OnClick="goConfirm('0','07')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type11" value="약용작물류"
                OnClick="goConfirm('0','19')" >
          </td>
        </tr>
        <tr class="board_line1px_e4e4e4">
          <td height="1" colspan="6" align="center"></td>
        </tr>
      <tr>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type12" value="양채류"
                OnClick="goConfirm('0','13')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type13" value="엽경채류"
                OnClick="goConfirm('0','10')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type14" value="인삼류"
                OnClick="goConfirm('0','18')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
```

```
hand" name="type15" value="잡곡류"
                OnClick="goConfirm('0','04')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type16" value="조미채소류"
                OnClick="goConfirm('0','12')" >
          </td>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type17" value="채소종자류"
                OnClick="goConfirm('0','15')" >
          </td>
        </tr>
        <tr class="board_line1px_e4e4e4">
          <td height="1" colspan="6" align="center"></td>
        </tr>
      <tr>
          <td height="23" align="center" >
              <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type18" value="특용작물류"
                OnClick="goConfirm('0','16')" >
          </td>
          <td height="23" align="center" >  </td>
          <td height="23" align="center" >  </td>
          <td height="23" align="center" >  </td>
          <td height="23" align="center" >  </td>
          <td height="23" align="center" >  </td>
```

나) JSP 코드

위의 디자인을 기반으로 하여 다음과 같이 JSP로 변환 하였다. 아래 JSP 코드는 Servlet과의 통신을 통해 데이터베이스에서 데이터를 전송 받아 위의 디자인과 같은 동일한 응답을 전송한다. JSP에서 HTML이 아닌 java 코드를 삽입할 경우 '<%' 태그로 java 코드의 시작을 표시하며 '%>' 태그로 종료를 표시한다.

```
<%
    List mainList=(List)request.getAttribute("mainList");
    int size1=mainList.size();
    for(int i=0;i<size1;i+=6){
%>
  <tr>
  <%
     for(int j=0;j<6;j++){
  %>
      <td height="23" align="center" >
      <%
          if ((i+j)<size1){
      %>
          <input type="submit"  class="samewidth_input" style="cursor:
hand" name="type<%=i+j%>" value="<%=((Grade) mainList.get(i+j)).getNa
me()%>"  OnClick="goConfirm('0','<%=((Grade)  mainList.get(i+j)).getCode
()%>')" >
    <%
       }else out.println("  ");
    %>
    </td>
    <%
              }
     %>
```

다) JSP 코드에 대한 사용자 응답 화면

위의 JSP로 작성되어진 화면을 사용자가 요청할 경우 아래와 같은 HTML로 되어진 응답 화면을 사용자에게 전송 한다.

```
<td height="23" align="center"><INPUT onClick="goConfirm('0','06')" type="submit" class="samewidth_input" value="과실류" name="type02"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','08')" type="submit" class="samewidth_input" value="과일과채류" name="type19"></td>
        <td height="23" align="center"><input onClick="goConfirm('0','14')" type="submit" class="samewidth_input" value="산채류" name="type82"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','11')" type="submit" class="samewidth_input" value="근채류" name="type32"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','03')" type="submit" class="samewidth_input" value="두류" name="type42"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','02')" type="submit" class="samewidth_input" value="맥류" name="type52"></td>
      </tr>
      <tr class="board_line1px_e4e4e4">
        <td height="1" colspan="6" align="center"></td>
      </tr>
      <tr>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','01')" type="submit" class="samewidth_input" value="미곡류" name="type62"></td>
```

```
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
17')" type="submit" class="samewidth_input" value="버섯류" name="type72
"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
14')" type="submit" class="samewidth_input" value="산채류" name="type83
"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
05')" type="submit" class="samewidth_input" value="서류" name="type92">
</td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
07')" type="submit" class="samewidth_input" value="수실류" name="type10
22"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
19')" type="submit" class="samewidth_input" value="약용작물류" name="ty
pe112"></td>
      </tr>
      <tr class="board_line1px_cccccc">
        <td height="1" colspan="6" align="center"></td>
      </tr>
      <tr>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
13')" type="submit" class="samewidth_input" value="양채류" name="type12
2"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
10')" type="submit" class="samewidth_input" value="엽경채류" name="type
132"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
18')" type="submit" class="samewidth_input" value="인삼류" name="type14
2"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
```

```
04')" type="submit" class="samewidth_input" value="잡곡류" name="type15
2"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
12')" type="submit" class="samewidth_input" value="조미채소류" name="ty
pe162"></td>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
15')" type="submit" class="samewidth_input" value="채소종자류" name="ty
pe17"></td>
      </tr>
      <tr class="board_line1px_cccccc">
        <td height="1" colspan="6" align="center"></td>
      </tr>
      <tr>
        <td height="23" align="center"><INPUT onClick="goConfirm('0','
16')" type="submit" class="samewidth_input" value="특용작물류" name="ty
pe18"></td>
        <td height="23" align="center"> </td>
        <td height="23" align="center"> </td>
        <td height="23" align="center"> </td>
        <td height="23" align="center"> </td>
        <td height="23" align="center"> </td>
      </tr>
```

다. 사용자 인터페이스 프로그램

디자인 되어진 화면을 기본으로 하여 사용자 인터페이스 화면을 구성하였다.

1) 첫 화면

시스템을 시작하면 나타나는 첫 화면이다.

그림 3.5.6 첫 화면

가) 로그인 화면

권한이 부여된 사용자를 인증하는 화면이다.

그림 3.5.7 로그인 화면

나) 관련홈페이지 화면

정부 기관과 농림/수산 관련 홈페이지에 대한 링크를 정리한 화면이다.

그림 3.5.8 관련 홈페이지 화면

다) 게시판 화면

게시물을 작성 관리하는 화면이다.

그림 3.5.9 게시판 첫 화면

2) 출하자

출하자는 정식 등록된 사용자로 농산물을 출하하는 농민이나 농협 담당 직원을 의미하며 접수증을 작성하는 주체이다. 아래 그림은 출하자가 인증되어졌을 경우 첫 화면이다.

그림 3.5.10 출하자 첫 화면

가) 등록상품리스트 화면

출하자가 등록한 접수증에 대한 리스트를 보여 준다. 리스트는 '전체', '금일', 출하준비중', '출하'와 같이 4범위로 조회해 볼 수 있으며 접수 번호를 클릭하면 해당 접수증을 수정 할 수 있다.

그림 3.5.11 등록 상품 리스트 화면

나) 접수증 수정 화면

입하 준비중인 접수증에 대해 수정이 가능하다. 입하되어진 접수증에 대해서는 우측 상단에 ' 입하 확인 되었습니다.'라는 메시지와 같이 수정이 불가능 하다.

그림 3.5.12 접수증 수정 화면

다) 상품 등록하기 화면

상품 등록화면은 신규 접수증을 작성하는 것으로 메뉴를 선택하게 되면 등록할 농산물에 대한 종류를 선택하도록 하였다.

그림 3.5.13 농산물 종류 선택

라) 등록 화면

앞 단계에서 선택되어진 농산물 종류에 따라 품목, 품종이 선택되어지도록 하였으며 10가지의 농산물이 등록되어지도록 하였다. 접수는 접수자 권한이 있는 사용자가 입하 확인을 한 경우 데이터가 입력되어지며 관리번호는 시스템에 의해 자동으로 부여되어진다. 접수번호는 등록 시도가 있을 때마다 증가하는 번호로써 본 시스템의 키 값으로 사용되어진다.

그림 3.5.14 등록화면

마) 출하자정보 수정

출하자의 개인 정보를 수정할 수 있도록 하는 화면이다.

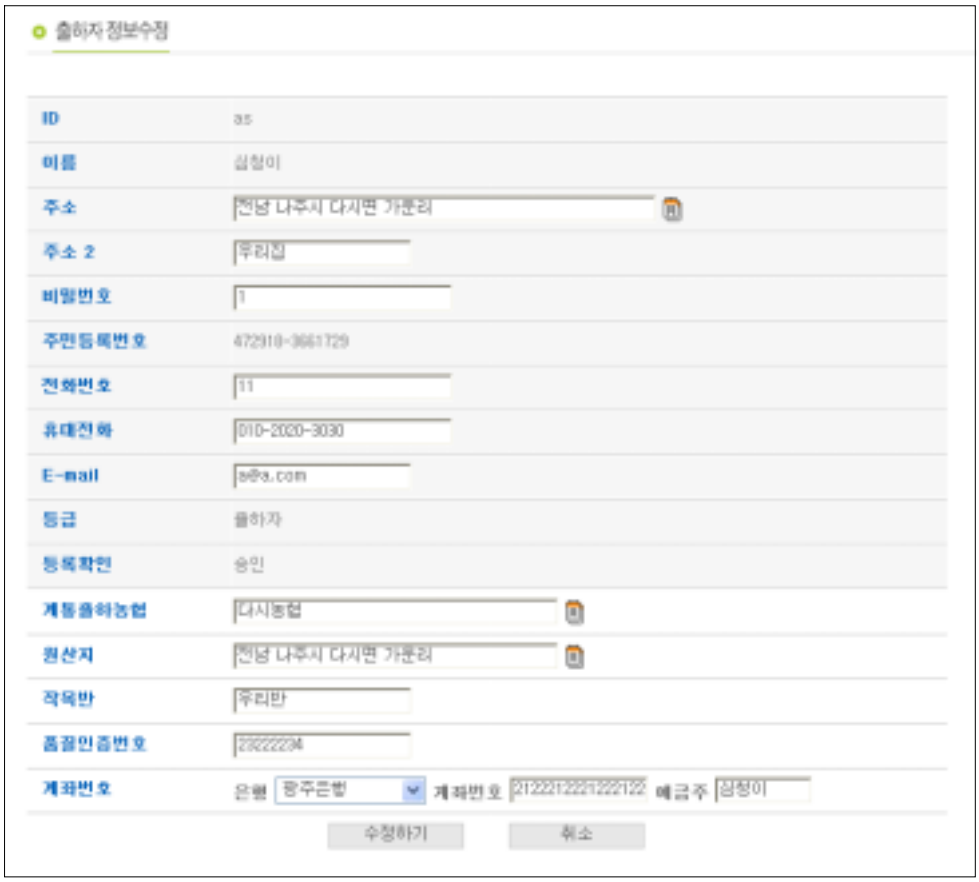

그림 3.5.15 출하자 정보 수정 화면

바) 현황보기 화면

현재 까지 등록되어진 농산물의 현황을 볼 수 있는 통계 화면으로 전체 통계 및 일자별 통계를 볼 수 있다.

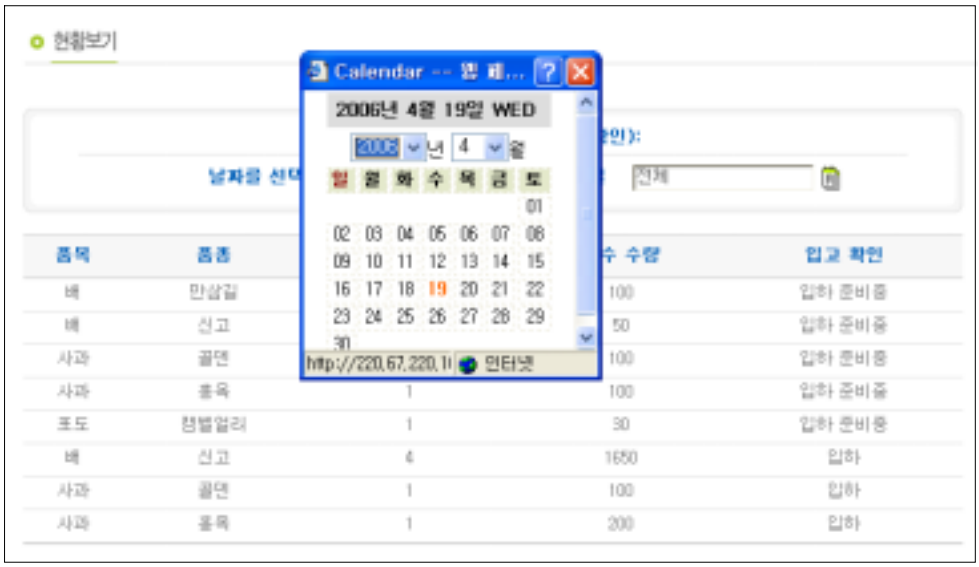

그림 3.5.16 현황보기 화면

3) 접수자

접수자는 정식 등록된 사용자로 출하자에 의해 작성되어진 접수증을 실제 입하되어진 농산물과 비교하여 입하 확인을 담당한다.

가) 접수자정보 수정

접수자의 개인정보를 수정할 수 있는 화면이다.

그림 3.5.17 접수자 정보 수정 화면

나) 접수증 검색 화면

입하확인 메뉴를 선택하면 나타나는 화면으로 입하확인 하고자 하는 접수증의 접수 번호 또는 출하자 ID로 검색할 수 있는 화면이다.

그림 3.5.18 접수증 검색 화면

다) 검색 리스트 화면

출하자 ID로 검색한 경우 아래와 같은 해당 접수증 리스트가 나타나며 접수번호를 클릭하면 해당 접수증에 대한 상세 정보로 이동한다.

그림 3.5.19 검색 리스트 화면

라) 입하확인 상세화면

입하 준비중인 상품과 비교해 보고 만일 농산물 정보에 변경이 있을 경우 '상품정보 수정' 버튼을 통해 농산물 정보를 변경하는데 이는 서버에 저장 관리되어

진다. 확인된 경우 '입하 확인' 버튼을 클릭하여 입하 확인을 한다.

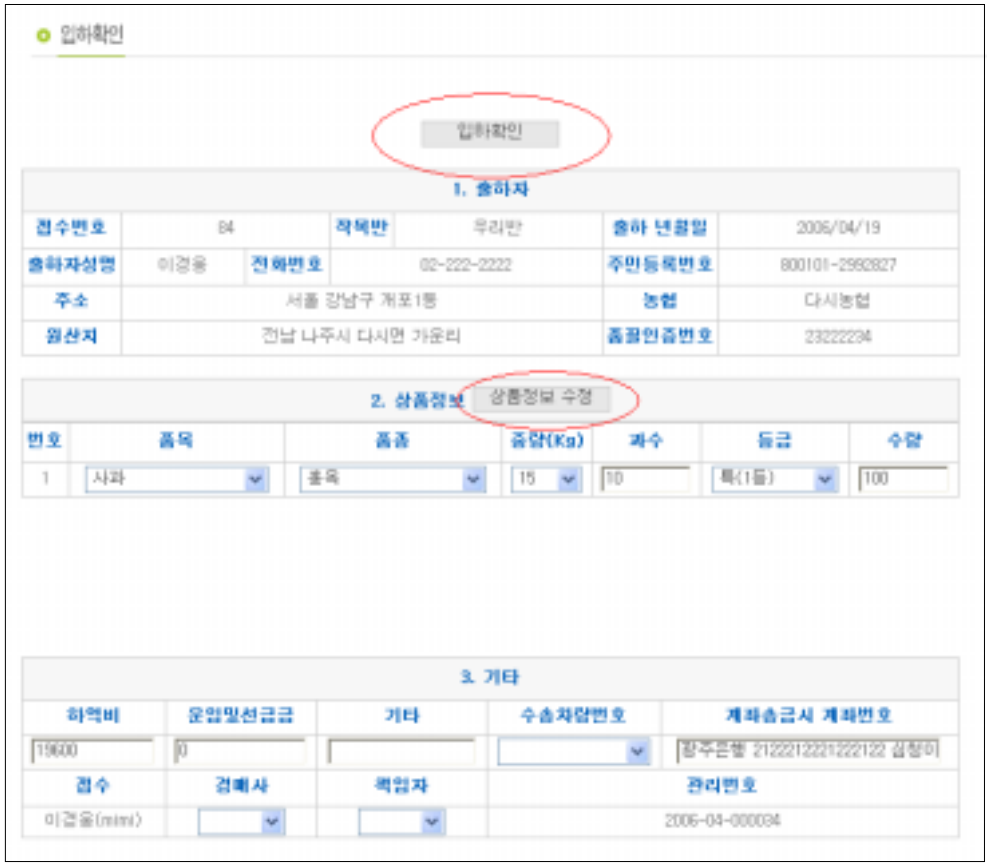

그림 3.5.20 입하 확인 상세 화면

4) 관리자

관리자는 시스템의 전반적인 관리를 담당하며 접수증의 내용에는 관여할 수 없으며 접수증의 삭제는 가능하다. 아래는 관리자 권한을 가진 사용자의 login이 성공하였을 경우 나타나는 첫 화면이다.

그림 3.5.21 관리자 첫 화면

가) 등록된 사용자 화면

기 등록되어진 사용자의 리스트를 보여준다. 본 시스템에서 사용자는 등록 요청을 하게 되고 관리자가 이를 확인하여 등록 확인을 해야 시스템을 사용할 수 있게 된다. 사용자 ID를 클릭하면 사용자의 정보를 변경할 수 있다.

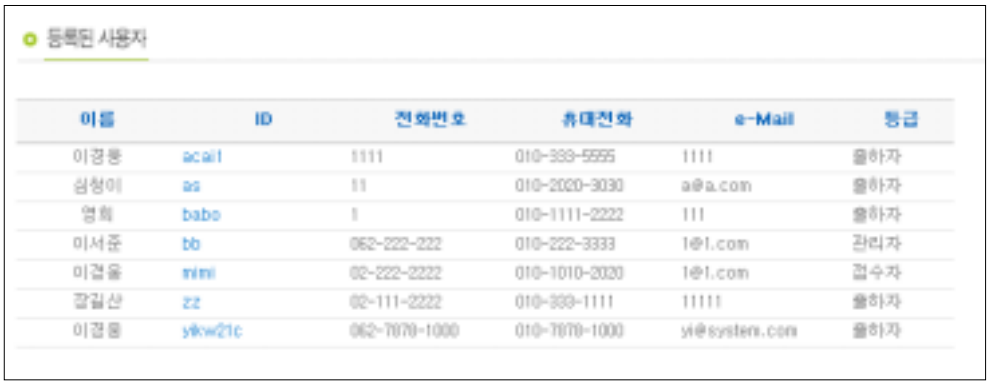

그림 3.5.22 등록된 사용자 리스트 화면

나) 관리자 정보 수정화면

리스트에서 ID를 선택하면 관리자 정보 수정화면이 나타난다. ‘ID’, ’이름‘, ’주민등록번호‘등은 변경할 수 없다.

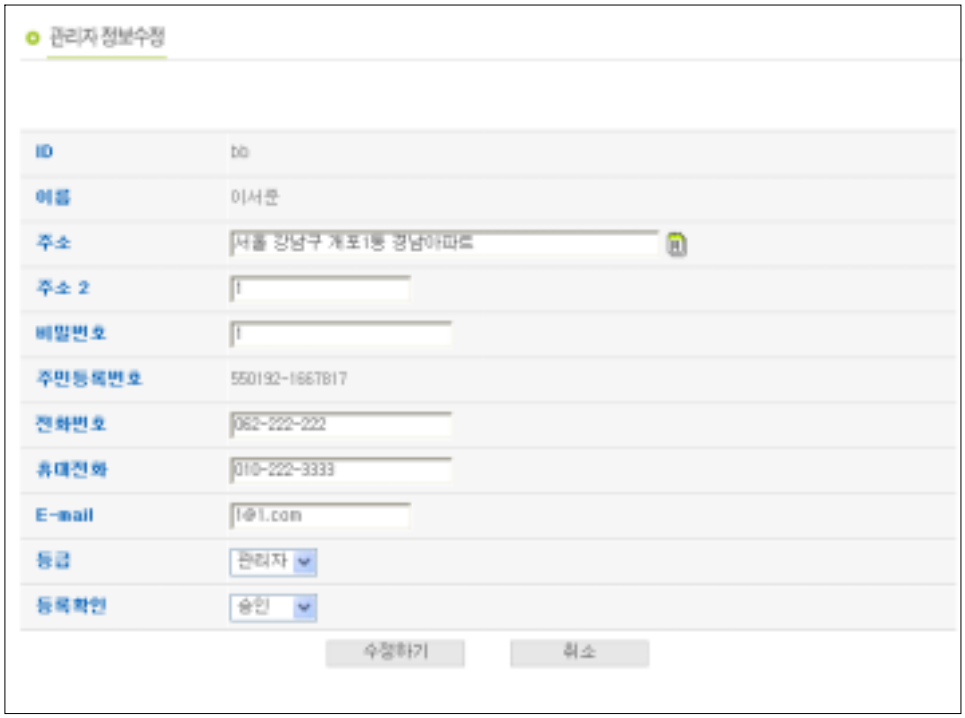

그림 3.5.23 관리자 정보 수정 화면

다) 등록요청사용자 리스트 화면

등록을 요청한 사용자의 리스트를 보여준다. ID를 클릭하면 해당 요청자의 정보가 나오고 관리자는 이를 확인하고 승인한다.

그림 3.5.24 등록요청 사용자 리스트 화면

라) 사용자 접수 확인 화면

등록 요청 사용자의 상세 정보 화면으로 등록확인 부분을 '미승인'에서 '승인'으로 변경하여 사용자 등록을 확인한다.

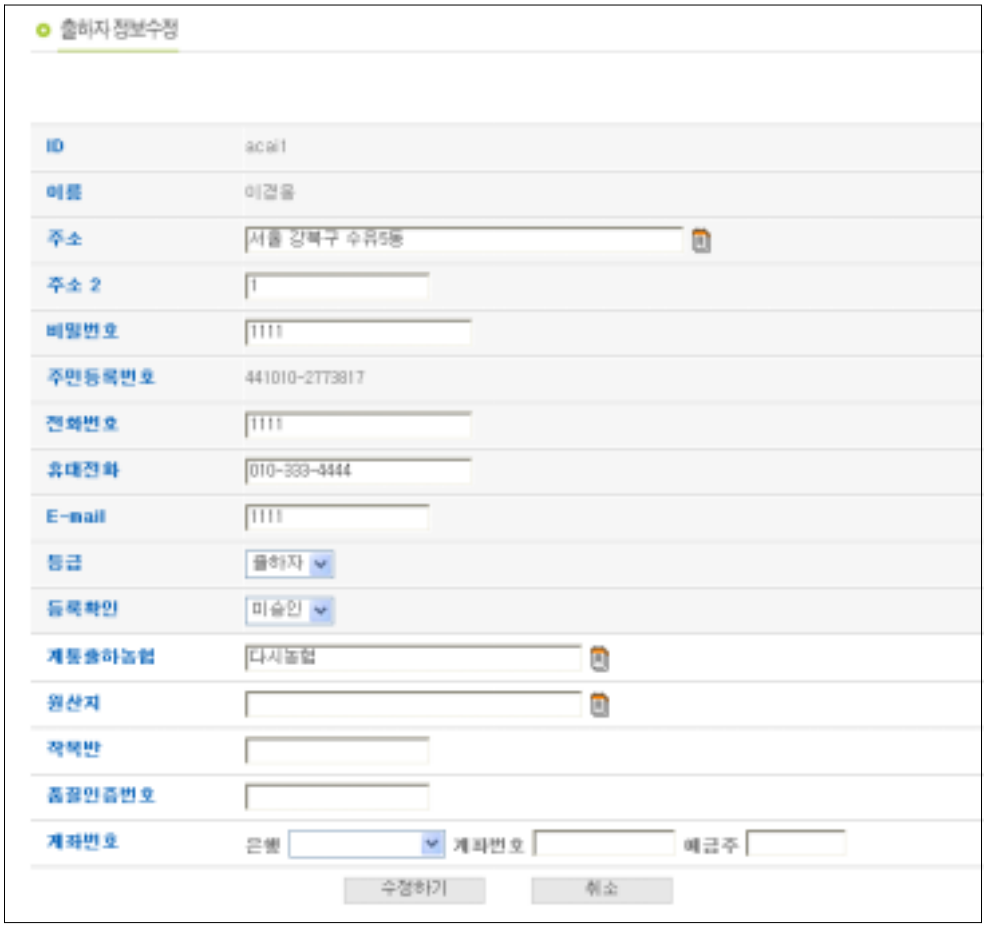

그림 3.5.25 사용자 접수 확인 화면

마) 접수 삭제 클릭

접수 삭제 메뉴를 선택하면 우선 접수증 검색화면이 나타난다. 접수번호를 알고 있는 경우에는 접수 번호로 검색이 가능하고 출하자 ID를 아는 경우에는 그 ID를 통해 등록되어진 접수증 리스트를 확인하고 삭제가 가능하게 된다. 접수증 삭제는 입하되어지지 않은 접수증에 대해서만 가능하고 이미 입하되어진 농산물에 대해서는 삭제를 할 수 없다.

그림 3.5.26 접수증 검색 화면

바) 접수삭제 리스트 화면

위에서 검색되어진 접수증의 리스트를 보여준다. 접수 번호를 클릭하여 선택한다.

그림 3.5.27 접수삭제 리스트 화면

사) 접수삭제 세부 화면

선택되어진 접수증의 세부 내용을 보여주는 화면이다. '접수 삭제하기' 버튼을 클릭하면 해당 접수증이 삭제되어진다. 접수증의 삭제는 입하 확인되지 않은 접수증에 대해서만 가능하다.

그림 3.5.28 접수 삭제 세부 화면

아) 접수 삭제 결과 화면

접수 삭제가 되었음을 알리는 화면이다.

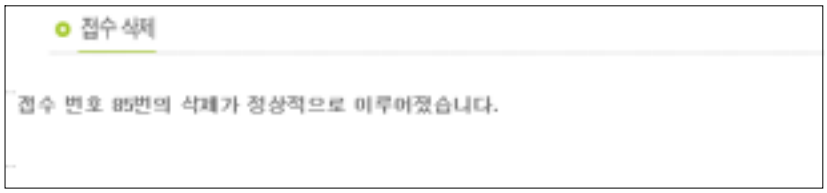

그림 3.5.29 접수 삭제 결과 화면

자) 코드관리

본 시스템에서 사용하는 코드를 관리하는 화면으로 변경 추가 또는 사용/미사용등을 이용하여 관리할 수 있다. 코드 관리에 사용되는 코드는 트럭 코드, 관리

자 코드, 경매사 코드 등이다.

그림 3.5.30 코드관리 화면

6. Mobile 기반의 클라이언트 시스템 구축

본 연구에서는 Windows PoketPC 2002 OS 기반의 PDA를 사용하여 프로그램을 작성하였다. 서버와의 통신은 HTTP를 기반으로 하는 SOAP을 사용하여 서버의 원격 객체를 사용하도록 하였으며 PDA에는 Business Logic을 최대한 배제하고 서버와의 통신을 통하여 운영되어지도록 하였다. 본 연구에서는 Mobile 기기로 PDA를 사용하였으며 그 해당 Application도 Windows PocketPC 2002에서 운영되어지는 것으로 국한 하였다.

가. PDA 기반 시스템 설계

PDA를 이용하여 시스템 사용자들의 공간적 제한을 최소화 한다.

1) PDA기반의 시스템

PDA기반의 시스템은 크게 PDA의 Internet Browser를 사용하여 시스템에 접근하는 것과 PDA 전용 Application 프로그램을 통한 접근으로 구분할 수 있다. Internet Browser를 통한 접근 기존의 Web 기반 시스템을 그대로 보여 줄 수 있으나 PDA가 갖는 화면의 제약 때문에 정상적인 시스템의 사용이 불가 하다. 그래서 본 연구에서는 PDA 전용 Application 프로그램을 구성하면서 Web 기반 시스템의 Business Logic을 그대로 사용할 수 있도록 SOAP 기반의 프로그램을 구성 하였다.

2) PDA 기반 Application

현재 PDA에는 다양한 운영체제가 사용되어지고 있으나 본 연구에서는 Microsoft사의 운영체제인 Pocket PC를 사용하는 PDA를 기준으로 Application을 개발하였다.

가) Pocket PC는 현재 버전이 Pocket PC 2003까지 출시되었는데 현재 Microsoft에서는 이들을 통 털어 Windows Mobile 2003이라는 이름으로 통합하고 있다. Pocket PC 2000과 Pocket PC 2002는 windows CE 3.X 운영체제 기반으로 구성된 제품이며 Pocket PC 2003은 Microsoft Windows CE .NET 4.2 운영체제를 기반으로 구성되어져 있다. Windows CE 3.0의 후속버전인 Window

s CE .NET은 Microsoft에서 출시한 최신의 eMbedded O/S이다. Windows CE.NET은 이전 버전의 견고한 기반 위에 확장 가능한 보안 네트워킹을 지원하고 더 빨라진 처리속도와 향상된 실시간 처리가 가능하다. 더욱 더 강력해진 PC와의 인터페이스 기능을 확장했으며 PDA에서 중요시되는 전원관리의 효율이 개선되었으며 .NET Framework 부분을 살펴보면, Common Language Runtime과 Base Class library 그리고, Data & XML과 ASP.NET & Web Services, Windows Forms로 구성되어져 있다.

나) .NET

.NET은 소프트웨어를 서비스로 운영하기 위한 Microsoft의 전략으로 새 서비스 생성 작업 및 작성을 위한 .NET 인프라 및 도구, 풍부한 클라이언트 사용을 위한 .NET 사용자 경험, 새 스마트 인터넷 장치 생성을 위한 .NET 장치 소프트웨어 등을 포함한다. NET Framework은 웹 서비스 및 기타 응용 프로그램의 작성, 배포 및 실행을 위한 환경으로 공용 언어 런 타임, Framework 클래스 및 ASP.NET의 세 가지 주요 부분으로 구성되어 있으며 .NET Compact Framework은 휴대폰, 향상된 텔레비전 등과 같은 장치로 일부 .NET Framework 기능을 가져오기 위한 것으로 NET Compact Framework는 Windows CE와 포함된 기타 운영 체제들에서 실행된다.

- .NET Framework 특징

- Application을 개발, 배치하는 것을 단순화, 현대화 시킨 새로운 플랫폼.
- 공개표준 프로토콜을 사용, 지원하는 인터넷 지향 어플리케이션 제작을 위해 설계.
- 풍부한 클래스 라이브러리를 제공.
- 모든 언어를 사용 가능하게 하는 언어 중립적인 플랫폼.
- COM, DLL으로 제작된 기존 컴포넌트를 상호 운용하도록 지원.
- 새로운 운영체제(Operating System)가 아닌, 관리된 런타임 환경을 의미.
- 독립적인 관리 환경인 Common Language Runtime을 통해 코드실행을 지원.

본 연구에서는 Windows Pocket PC 2002 .Net Framework 기반에 C#을 이용하여 PDA 사용자를 위한 Application을 작성하였다.

나. WSDL을 이용한 클래스 구현

서버 프로그램에서 정의되어진 WSDL을 기준으로 .NET Framework에서 사용되어진 Interface 클래스를 C#으로 작성하였으며 Application Program은 접수자용 프로그램과 출하자용 프로그램 두 가지로 작성하였다. 두 사용자를 위한 프로그램의 WSDL은 동일하다.

```
namespace SmartDeviceApplication2.WebService {
    using System.Diagnostics;
    using System.Xml.Serialization;
    using System;
    using System.Web.Services.Protocols;
    using System.ComponentModel;
    using System.Web.Services;

    /// <remarks/>
    [System.Diagnostics.DebuggerStepThroughAttribute()]
    [System.ComponentModel.DesignerCategoryAttribute("code")]
    [System.Web.Services.WebServiceBindingAttribute(Name="userAuthSo
apBinding", Namespace="http://userAuth.webservice")]
    [System.Xml.Serialization.SoapIncludeAttribute(typeof(Grade))]
    [System.Xml.Serialization.SoapIncludeAttribute(typeof(CommManage))]
    [System.Xml.Serialization.SoapIncludeAttribute(typeof(RceiptProductSer
v))]
    [System.Xml.Serialization.SoapIncludeAttribute(typeof(ReceiptProduct))]
    public class UserAuthIFService : System.Web.Services.Protocols.Soap
HttpClientProtocol {

        /// <remarks/>
        public UserAuthIFService() {
```

```
            this.Url = "http://220.67.220.162:8080/regManage/services/userAuth";
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("login1Return")]
        public int login1(string in0, string in1) {
            object[] results = this.Invoke("login1", new object[] {
                        in0,
                        in1});
            return ((int)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult Beginlogin1(string in0, string in1, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("login1", new object[] {
                        in0,
                        in1}, callback, asyncState);
        }

        /// <remarks/>
        public int Endlogin1(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((int)(results[0]));
        }
```

```
        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getUserReturn")]
        public User getUser(string in0) {
            object[] results = this.Invoke("getUser", new object[] {
                        in0});
            return ((User)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetUser(string in0, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("getUser", new object[] {
                        in0}, callback, asyncState);
        }

        /// <remarks/>
        public User EndgetUser(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((User)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
```

```
        [return: System.Xml.Serialization.SoapElementAttribute("getUserERe
turn")]
        public UserE getUserE(string in0) {
            object[] results = this.Invoke("getUserE", new object[] {
                        in0});
            return ((UserE)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetUserE(string in0, System.Asy
ncCallback callback, object asyncState) {
            return this.BeginInvoke("getUserE", new object[] {
                        in0}, callback, asyncState);
        }

        /// <remarks/>
        public UserE EndgetUserE(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((UserE)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getRegNu
mReturn")]
        public string[] getRegNum(string in0) {
            object[] results = this.Invoke("getRegNum", new object[] {
                        in0});
```

```
            return ((string[])(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetRegNum(string in0, System.
AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("getRegNum", new object[] {
                        in0}, callback, asyncState);
        }

        /// <remarks/>
        public string[] EndgetRegNum(System.IAsyncResult asyncResult)
{
            object[] results = this.EndInvoke(asyncResult);
            return ((string[])(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getItemLis
tReturn")]
        public ReceiptProduct[] getItemList(string in0) {
            object[] results = this.Invoke("getItemList", new object[] {
```

이하 부록2 참조

다. Web Browser를 통한 PDA 사용자 프로그램 구현

1) 개요

PDA기반의 시스템은 크게 PDA의 Internet Browser를 사용하여 시스템에 접근하는 것과 PDA 전용 Application 프로그램을 통한 접근으로 구분할 수 있다. Internet Browser를 통한 접근 기존의 Web 기반 시스템을 그대로 보여 줄 수 있다. Windows OS 기반의 PDA는 Windows CE와 Windows Pocket PC 두 종류로 나눌 수 있는데 Windows CE를 사용 할 경우에는 기존의 Internet Browser를 사용하는 것과 같이 시스템을 사용할 수 있으나 Windows Pocket PC를 사용할 경우에는 Windows Browser의 크기가 작아서 시스템을 사용하는데 많은 불편이 있게 된다. 인터넷의 발달은 개인휴대단말기(PDA)에 있어서도 모바일 인터넷이라는 새로운 변화를 가져왔다.

초기 PDA는 통신 기능을 수용하기 어려웠다. 그러나 정보기술 발달과 인터넷을 통한 정보 획득에 대한 수요는 PDA 에서 인터넷을 통한 통신 기능의 탑재를 가능하게 했다. PDA로 인터넷을 이용하는 방법은 오프라인 브라우저(Offline Browser), 무선랜, CDMA 등 크게 세 가지로 나뉜다.

오프라인 브라우저는 PDA에서 인터넷 정보를 활용하기 위한 초기 수단으로 웹 페이지를 PDA에 그대로 저장하는 방식이다. CDMA를 이용한 인터넷 통신은 PDA에 휴대폰 기능이 있는 기기를 결합함으로써 휴대폰과 통신기기로 병행 사용이 가능하다. 언제 어디서나 인터넷에 접속해 원하는 정보를 얻을 수 있다는 장점 이 있다. 그러나 CDMA 인터넷은 인터넷 사용시 휴대폰 사용 시간과 동일한 요금이 책정되므로 사용료가 비싸고 속도가 느리다는 단점이 있다. 무선랜을 이용한 인터넷은 최근 개발된 것으로 사용료가 저렴하면서 빠른 속도로 사용할 수 있다. 무선랜을 이용하기 위해서는 인터넷 접속을 가능하게 하는 AP(Access Point)라는 장치가 필요하며 이 AP의 유효한 반경 이내에서 PDA의 무선 랜 카드를 이용해 인터넷을 사용할 수 있다. PDA로 인터넷을 사용하기 위해서는 기존 인터넷 사이트와는 차별된 전용 서비스가 필요하다. 전용 사이트는 갈수록 늘어나고 있으며 이미 많은 기업이 PDA를 위한 전용 사이트 서비스를 시작하고 있다.

2) Windows CE와 Pocket PC

Windows CE 는 Microsoft의 임베디드 시스템을 위한 플랫폼의 이름이다. 여기서 말하는 임베디드라는 것은 컴퓨터에 장착된 CPU처럼 컴퓨팅을 하기 위해 만들어 진 것이 아니라 특정한 기능을 지원하기 위해 제작 사용되는 전용 칩을 의미한다. 반면에 PocketPC 는 MS에서 지정하고 하드웨어 규격을 MS에서 테스트 받은 PDA 하드웨어에 최적화된 Windows CE 운영체제를 가리킨다.

WindowsCE 와 PocketPC는 소프트웨어 사용면에서 달라진다. 프로그램 개발에 사용되는 API는 PocketPC 보다 WindowsCE가 더 많기 때문이다. 따라서 PocketPC 기반에서 개발되어진 프로그램은 Windows CE 운영체제에서 사용할 수 있지만, Windows CE 기반에서 개발되어진 프로그램은 PocketPC에서 사용할 수 없는 경우가 발생한다. PocketPC는 Windows CE에 호환되지만, Windows CE는 PocketPC에 완전히 호환되지는 않는다.

3) 본 연구에서는 Windows CE 기반의 PDA 사용자의 경우 Internet Browser를 사용하여 시스템을 사용하고 Windows Pocket PC 기반의 PDA 사용자의 경우에는 별도의 Application Program을 사용하도록 하였다. 아래 그림은 Windows CE 기반의 PDA를 사용하여 시스템에 접근하는 화면을 보이는 것이다. 그림에서와 같이 Windows CE 기반의 PDA를 사용할 경우에는 PC에서 사용하는 화면과 동일한 화면에서 시스템을 사용할 수 있다.

4) PDA를 사용한 Browser 화면

Windows CE 기반의 PDA는 PC에서 사용하는 Browser에 보이는 화면과 동일한 화면을 제공한다. 따라서 별도의 program 없이도 시스템을 사용할 수 있게 된다.

가) 첫 화면

아래 그림은 PDA의 Internet Browser를 사용하여 실행하여 실행되어진 시스템의 첫 화면이다.

그림 3.6.1 windows CE 기반의 첫화면

나) Login 화면

아래 그림은 PDA의 Internet Browser를 사용하여 실행하여 실행되어진 시스템 Login 화면이다.

그림 3.6.2 windows CE 기반의 Login 화면

다) 접수자 화면

아래 그림은 PDA의 Internet Browser를 사용하여 실행하여 실행되어진 접수자의 첫 화면이다.

그림 3.6.3 windows CE 기반의 접수자 화면

다) 출하자 화면

아래 그림은 PDA의 Internet Browser를 사용하여 실행하여 실행되어진 출하자의 첫 화면이다.

그림 3.6.4 windows CE 기반의 출하자 화면

마) 관리자 화면

아래 그림은 PDA의 Internet Browser를 사용하여 실행하여 실행되어진 관리자의 첫 화면이다.

그림 3.6.5 windows CE 기반의 관리자 화면

라. Application Program을 통한 PDA 사용자 프로그램 구현

1) 개요

본 연구에서는 PDA용 Application Program 개발을 위해 .Net Framework 기반의 C#을 사용하였다. Mobile Program의 특성상 여러 Form을 사용하지 않고 단일 Form에서 여러 Panel을 사용하여 화면 이동을 시행하였다. 다음은 농산물 등록과 입하 관리 Application의 디자인 화면을 보여주고 있다.

PDA의 Application에는 Business Logic이 포함되어 있지 않고 사용자 Interface 만이 구현되어져 있으며 Business Logic은 Internet Browser 사용자가 이용하는 것과 동일하게 Server에 저장되어진 Logic을 사용한다.

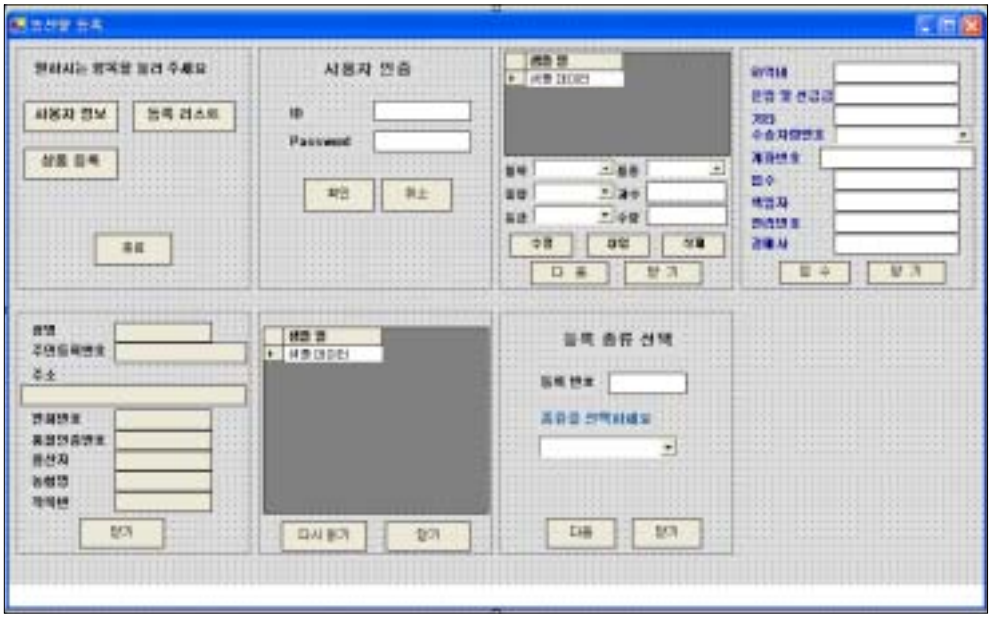

그림 3.6.6 농산물 등록 Application 디자인

그림 3.6.7 입하 관리 Application 디자인

2) 농산물 등록 Application 흐름도

출하자가 농산물을 등록하고 관리하는 Application 이다.

가) 출하자 농산물 등록 Application은 사용자 인증을 하게 되는데 출하자가 아닌 관리자 또는 접수자로 인증 요청을 하게 되면 아래 그림과 같은 메시지를 보이며 사용할 수 없음을 보여준다.

나) 출하자 정보는 변경할 수 없으며 변경은 홈페이지를 통해서 시행한다.

다) '등록 리스트'는 해당 출하자의 등록 농산물 리스트를 보여준다. 등록리스트에서는 리스트의 내용을 선택하여 볼 수 있도록 항목 선택이 있어 필요한 부분만을 볼 수 있도록 하였다.

라) 농산물 내역 및 기타정보 수정은 각각 시행 관리되어지도록 하였다.

마) '농산물 등록'은 우선 농산물의 종류를 선택하고 농산물의 내용을 등록하도록 하였다.

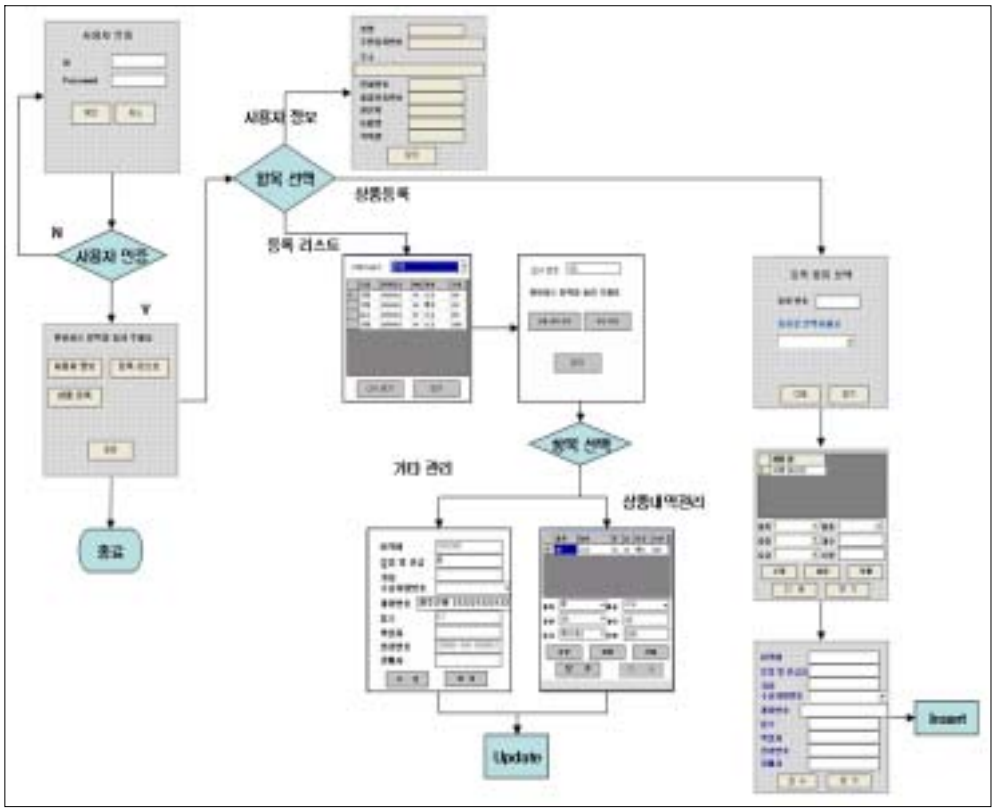

그림 3.6.8 농산물 등록 Application 흐름도

3) 입하 관리 Application 흐름도

접수자에 의해 농산물의 입하를 확인하는 Application 이다.

가) 출하자 농산물 등록 Application은 사용자 인증을 하게 되는데 접수자가 아닌 관리자 또는 출하자로 인증 요청을 하게 되면 아래 그림과 같은 메시지를 보이며 사용할 수 없음을 보여준다.

나) 인증이 되고나면 출하자의 ID로 출하 내역을 검색 하도록 하였다.

다) 메뉴는 '사용자 정보', '농산물 정보', '입하확인'으로 되어있다.

라) '사용자 정보'는 확인만 할 수 있으며 수정을 할 수 없도록 하였다.

마) '농산물 정보'를 선택하면 해당 출하자가 등록한 농산물중 입하 준비 중인 농산물의 리스트를 보여주며 리스트 중 특정 농산물을 선택하여 수정이 가능하도록 하였다. 수정되어진 농산물 정보는 Internet Browser를 이용한 시스템의 Business Logic을 사용함으로 변경 정보가 DataBase에 저장되어진다.

바) ‘입하 확인’을 선택하면 해당 접수증에 대한 상태가 입하 준비 중에서 입하 확인으로 변경되어진다.

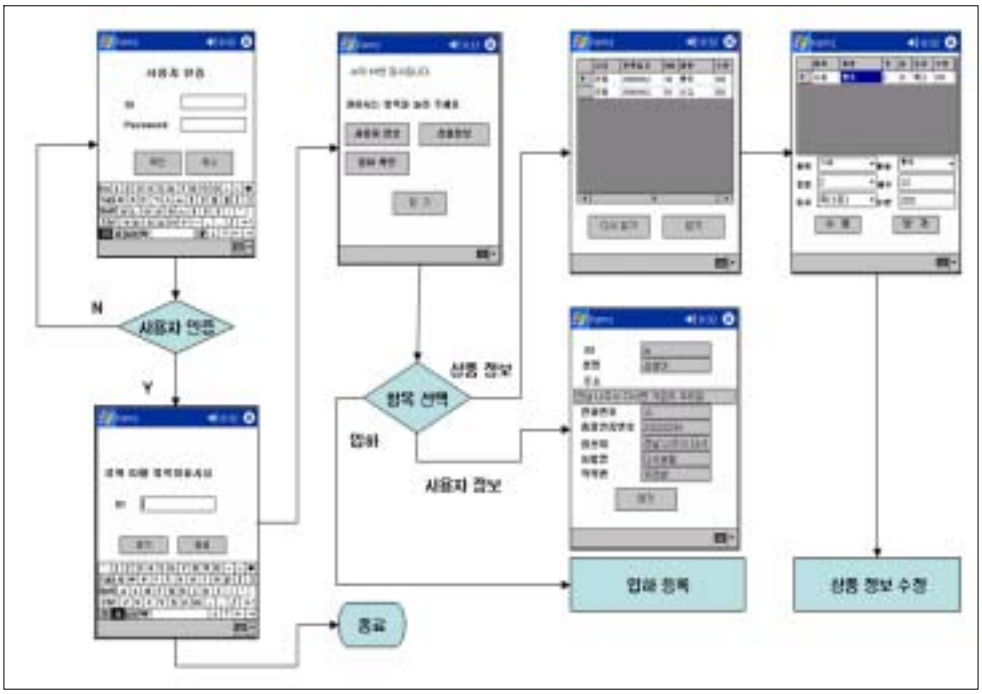

그림 3.6.9 입하 관리 Application 흐름도

4) 농산물 등록 Application 실행

PDA를 이용해 출하자가 농산물을 등록 할 때 사용하는 Application이다.

가) 사용자 인증

인증 화면이 실행되면 입력메뉴를 바로 실행하여 사용자가 직접 실행하지 않아도 되도록 하였다. 본 Application은 출하자를 대상으로 만들어진 Application으로 관리자나 접수자의 계정으로 실행할 경우 실행되지 않는다. 인증 확인이 성공하는 경우에는 주 메뉴로 이동하고 실패하는 경우에는 사용자 인증 화면으로 다시 이동한다. 확인 버튼을 클릭하면 인증을 시도하고 취소 버튼을 클릭하면 Application을 종료한다.

그림 3.6.10 사용자 인증

나) 주 메뉴화면

농산물 등록 Application의 주 메뉴는 크게 '사용자 정보', '등록 리스트', '농산물 등록'으로 이루어져 있으며 해당 메뉴를 클릭하면 실행되어진다. 종료 버튼을 클릭하면 Application을 종료한다.

그림 3.6.11 주 메뉴화면

다) 사용자 정보

'사용자 정보' 화면에서는 사용자의 일반 정보를 보여준다. 그러나 수정은 불가 하도록 되어져 있으며 수정이 필요할 경우 Internet Browser에서 수정하여야 한다. 닫기 버튼을 클릭하면 주 메뉴로 이동한다.

그림 3.6.12 사용자 정보

라) 등록 리스트

현재 Login한 출하자가 등록한 농산물의 내역을 보여준다. 등록 리스트 화면이 실행되면 당일 등록 농산물에 대한 리스트가 나오면 출하자는 리스트 조건을 선택하여 여러 형태로 조회해 볼 수 있다. 리스트 조건은 '전체', '금일', '입하 준비중' , '입하' 4가지로 구성되어져 있다. 다시 읽기 버튼을 클릭하면 리스트의 내용을 서버에서 다시 읽어와 보여주고 닫기 버튼을 클릭하면 주 메뉴 화면으로 이동한다.

그림 3.6.13 등록 리스트

마) 접수 정보 수정

등록 리스트에서 농산물을 선택하면 해당 접수에 대해 수정할 수 있도록 한다. 이미 입하가 되어진 접수에 대해서는 수정을 할 수 없다. 접수 정보 수정 메뉴는 ‘ 농산물 내역 관리’, ‘ 기타 관리’로 이루어져 있다. 닫기 버튼을 클릭하면 등록 리스트 화면으로 이동한다.

그림 3.6.14 접수 정보 수정

바) 농산물 내역 관리

농산물 내역 관리는 해당 접수에 기재되어진 농산물들의 목록을 보여주면 출하자의 필요에 따라 변경할 수 있도록 되어져 있다. 변경 내용으로는 단순 수정과 새로운 농산물의 삽입 기존 농산물의 삭제로 되어져 있으며 수정, 삭제, 삽입을 실행하면 '적용' 버튼이 활성화 되어지며 적용 버튼을 실행하지 않으면 변경사항이 저장되어지지 않는다. 수정 버튼을 클릭하면 내용 수정이 이루어지고 삽입 버튼을 클릭하면 아래에서 작성되어진 내용이 리스트에 삽입되어지고 삭제 버튼을 클릭하면 선택되어진 리스트의 내용을 삭제한다. 적용 버튼을 클릭하면 리스트의 변경 사항이 서버에 저장되어지며 닫기 버튼을 클릭하면 접수정보 수정 메뉴화면으로 이동한다.

그림 3.6.15 농산물 내역 관리

사) 기타 관리

접수증의 기타 항목을 수정할 수 있다. 기타 항목 중 '하역비', '관리번호', '접수' 등은 수정할 수 없다. 하역비와 관리번호는 시스템에 의해 자동 생성되어지는 항목이며 접수는 접수자가 접수 하였을 경우 시스템에서 해당 접수자의 ID를 입력함으로 출하자가 수정할 수 없다. 수정을 클릭하면 변경 내용이 Server에 저장되어지며 닫기 버튼을 클릭하면 접수정보 수정 메뉴화면으로 이동한다.

그림 3.6.16 기타 관리

아) 농산물 등록 메뉴

농산물등록을 선택하면 아래와 같은 농산물등록 메뉴가 나타난다. 농산물등록 메뉴의 접수 번호는 시스템에서 부여하는 것으로 농산물등록을 선택하면 무조건 접수 번호를 생성한다. 농산물의 종류는 DataBase에서 종류를 가져오며 다음을 클릭하면 다음 단계로 넘어가며 닫기를 클릭하면 주 메뉴 화면으로 이동한다.

그림 3.6.17 농산물 등록 메뉴

자) 농산물 등록

농산물내역을 삽입, 수정, 삭제하는 부분이다. 해당 버튼을 클릭 하여도 Server에 저장되어지지 않으며 다음 단계까지 완료되어야 저장되어진다. 다음을 클릭하면 다음 단계로 진행하며 닫기를 클릭하면 주 메뉴로 이동한다.

그림 3.6.18 농산물 등록

차) 기타 입력

기타 입력은 접수증의 기타 내용을 입력하는 화면으로 하역비 관리번호 등은 시스템에 의해 자동으로 처리되기 때문에 입력이 불가하다. 접수버튼을 클릭하면 Database에 저장되고 닫기를 클릭하면 저장하지 않고 주 메뉴로 이동한다.

그림 3.6.19 기타 입력

5) 입하 관리 Application

입하 관리 Application은 출하자가 등록한 농산물의 내용을 접수자가 확인하고 입하 여부를 결정한다.

가) 사용자 인증

입하 관리 Application은 접수자가 입하 여부를 관리하는 Application으로 출하자와 관리자의 계정으로는 사용할 수 없다. 사용자 인증 화면이 실행되면 입력 메뉴가 자동으로 실행되게 하여 사용자가 입력 메뉴를 직접 클릭하는 불편을 해소 하였다. 사용자 계정이 확인되면 검색 화면으로 이동하고 인증이 실패 하였을 경우에는 Message를 표시하고 다시 인증화면으로 이동한다.

그림 3.6.20 사용자 인증

나) 검색

사용자 인증이 성공하면 나타나는 화면으로 출하자 ID를 이용하여 대상 출하자의 해당 접수를 검색한다. 검색은 입하 준비 중인 접수에 대해서만 이루어진다. 검색이 완료되면 검색 사용자가 등록한 리스트를 보여주는 리스트 화면으로 이동하고 종료를 클릭하면 Application이 종료된다.

그림 3.6.21 검색

다) 검색 리스트

검색되어진 사용자가 등록한 접수증 중 출하 준비 중인 접수증의 리스트를 보여준다. 리스트의 항목을 클릭하면 해당 접수증의 주 메뉴로 이동하고 다시 읽기를 클릭하면 리스트의 내용을 Server에서 다시 읽어 오고 닫기를 누르면 검색 화면으로 이동한다.

그림 3.6.22 검색 리스트

라) 주 메뉴

검색 되어진 사용자의 입하 관리 Application의 주 메뉴는 '사용자 정보', '농산물 정보', '입하 확인'으로 구성되어져 있으며 닫기를 클릭하면 검색 리스트 화면으로 이동한다.

그림 3.6.23 주 메뉴

마) 사용자 정보

해당 접수증의 사용자 정보를 보여주며 수정은 불가하다. 닫기 버튼을 클릭하면 주 메뉴로 이동한다.

그림 3.6.24 사용자 정보

바) 농산물 정보

해당 접수증의 농산물 정보를 보여주고 수정할 수 있다. 농산물 리스트에서 특정 항목을 클릭하며 해당 항목에 대한 상세 정보를 보여주며 입하 준비 중인 농산물과 내용이 다를 경우 수정할 수 있으며 수정 내용은 서버에 저장 관리되어진다. 변경 사항이 발생하면 수정 버튼이 활성화 되어지고 수정하는 기능을 수행하며 닫기를 클릭하면 주 메뉴로 이동한다.

그림 3.6.25 농산물 정보

7. 시작품 제작

가. 시작품 제작

본 연구에서는 “농산물 상장 등록 시스템”을 구축하여 정상적인 운영을 확인하였다. Web Server와 Web Application Server를 동일한 Server에 설치하였고 Data base는 별도의 Server에 설치하였다. Mobile 기기로는 PDA를 이용하였으며 PDA의 통신을 위해 IEEE 802.11g 기반의 Access Point를 이용한 무선 LAN방법과 휴대폰 통신망인 CDMA망을 이용한 방법에 대해 테스트를 시행하였다. PC를 이용한 시스템 이용은 공히 ADSL을 이용하였다.

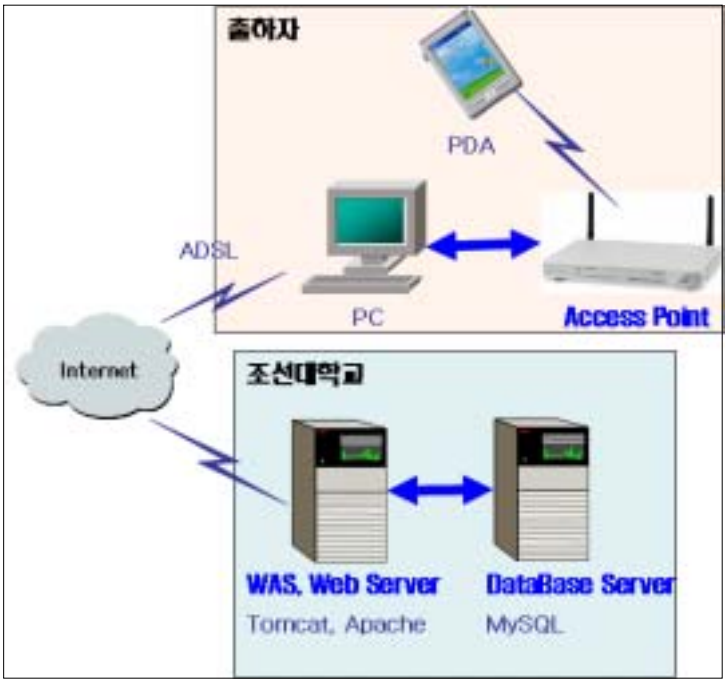

그림 7.1.1 시제품 시스템 구성도(Access Point)

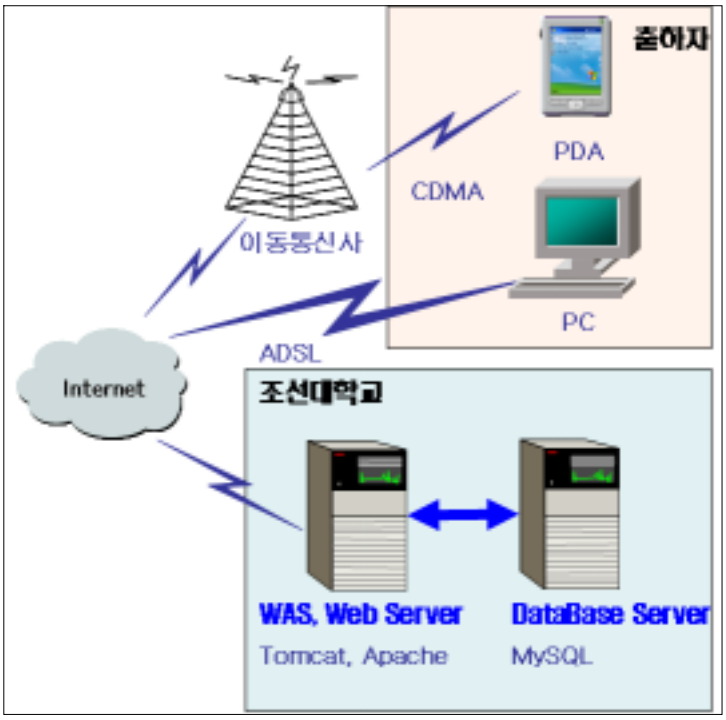

그림 7.1.2 시제품 시스템 구성도(CDMA 망)

1) 서버 설치 및 개발

시스템의 구축을 위해 다음과 같이 Solution을 설치하고 개발하였다.

가) DataBsae는 Mysql 5.0을 사용하였으며 Tomcat은 5.5.9버전을 사용하였고 Apache는 2.0 버전을 사용하였다.

나) 개발 언어로는 Java 1.5 기반에서 JSP 및 Servlet, javaBeans를 사용하였으며 개발 Framework은 Struts를 사용하였다.

다) Mobile 개발은 Microsoft사의 VisualStudio2002 기반에 언어는 C#을 이용 개발하였다. 시험운영은 PDA를 이용하였으며 시험운영에 사용되어진 PDA의 OS는 Windows PocketPC 2002이다.

2) 시험 운영

본 시스템의 정상적인 운영을 확인하기 위해 다음과 같이 시험 운영하였다.

가) 시스템에 필요한 Code 데이터를 수집 입력하였다.

나) PC의 Internet Browser(MicroSoft사의 explore)를 이용하여 출하입력 및 입고

확인 등 모든 기능을 수행하였다.

다) PDA를 통하여 출하입력 및 입고 확인 등 모든 기능을 수행 하였다.

라) PDA를 통해 입력 또는 수정되어진 데이터를 PC의 Internet Browser를 통해 확인 하였으며 또한 PC의 Internet Browser를 통해 입력 또는 수정된 데이터를 PDA를 통해 확인함으로써 시스템의 정상적 운영을 확인하였다.

마) 다수 사용자가 PDA와 PC의 Internet Browser를 통해 시스템을 사용하도록 하여 시스템의 안정적인 운영을 확인하였다.

바) PDA를 이용한 시스템 사용을 위해 통신 방식을 휴대폰망인 CDMA망을 이용한 테스트와 3COM사의 Access Point를 통한 무선 LAN을 이용한 방법 모두에 대해 테스트를 수행 하였으며 정상적인 작동을 확인하였다.

제 4 장 목표달성도 및 관련분야에의 기여도

제 1 절 연구개발 목표의 달성도

연 구 개 발 목 표	달 성 도
1. 업무 분석	완 료
2. DBMS 구축	완 료
3. 서버 시스템 설계 및 구축	완 료
4. 서버 프로그램 설계 및 구현	완 료
5. Web Browser 기반의 클라이언트 시스템 구축	완 료
6. Mobile 기반의 클라이언트 시스템 구축	완 료
7. 시작품 제작	완 료

제 2 절 관련분야에의 기여도

1. 기술적 측면

가. 정보화 선진국인 우리나라의 강점을 살려 인터넷과 유무선 통신망을 이용한 출하자 정보 입력 시스템을 구축.

나. 원거리에서 정보를 전송하여 상장신청 양식에 입력할 수 있는 기술을 개발.

다. 컴퓨터에 상장신청에 필요한 정보를 생성시키고 인터넷 통신망을 이용하여

정보를 원격 입력하는 시스템을 개발.

라. 소형 컴퓨터인 PDA에 정보를 생성하고 유 무선통신 네트웤 기반을 이용하여 정보를 원격 입력하는 시스템을 개발.

2. 경제・산업적 측면

가. 상장 정보 입력을 위한 작업시간과 업무량을 줄임으로써 도매시장의 생산성 향상.

나. 상장 시 작성하는 접수증 양식의 표준화를 유도.

다. 인건비 절감으로 인한 수수료의 인하 요인을 제공.

3. 사회・문화적 측면

가. 매일 반복되는 경매 준비를 위한 긴장감을 해소할 수 있다.

나. 접수증에 농산물의 정보를 작성할 때 수기가 아닌 컴퓨터 활자를 사용함으로써 기록 오류를 방지할 수 있다.

4. 농산물 상장 방법에 대한 건의

정보화 인프라가 잘 구축된 정보화 선진국으로서 강점을 활용하여 도매법인에서 상장을 위해 농산물 정보를 입력하는 시간과 업무량을 대폭 줄이는 효과를 얻을 수 있는 방법이고, 현행 도매시장의 운영 방법에서 개선해야 할 부분을 보완하면 일정 규모를 갖춘 농가를 대상으로 본 방식이 채택될 수 있을 것으로 전망된다.

제 5 장 연구개발결과의 활용계획

1. 본 연구에 의해 개발된 핵심기술은 산지에서 출하자가 자신의 농산물에 대한 상장정보를 입력하는 시스템이다. 컴퓨터를 이용하여 상장 신청서에 수록할 상장 정보들을 도매법인의 서버로 전송하는 시스템과, 모바일을 이용하여 유무선통신망으로 상장 정보를 도매법인의 서버에 전송하는 시스템 기술이다.
2. 활용분야는 도매시장에서 상장정보를 입력하는 분야에 적용할 수 있다.
3. 활용유형은 주관기관 명의로 특허를 출원하고, 도매법인에 기술을 이전하여 도매법인의 애로사항을 해결할 수 있도록 한다.
4. 추가 기술은 이전 받은 도매법인이 현장감각에 맞도록 추가 개발한다.

제 6 장 연구개발과정에서 수집한 해외과학기술정보

해당사항 없음

제 7 장 참고문헌

1. Advanced JSP 사이텍 미디어 David M.Geary저/권기현 역, 2001
2. Java 분산프로그래밍 홍릉과학미디어 김일민, 박재희 공저, 2205
3. Java How To Program 피어슨에듀케이션코리아 하비 디텔,폴 디텔 공저/민현기 역, 2005
4. My SQL Web DB 실무 프로그래밍 혜지원 이법기 외, 2001
5. My SQL Language Reference(Paperback) OMYSQL Paul DuBois, 2002
6. 웹디자인 기획 및 실무, 출판사: 이한출판사 김소영, 2003
7. Best 웹디자인 활용 테크닉 정보문화사 고영자, 최수영 공저, 2001
8. Java Design Patterns : A Tutorial, Addision-Wesley Professional, James William Cooper, 2000
9. 자바를 이용한 웹 서비스 구축: XML・SOAP・WSDL・UDDI의 이해, 인포북, Simeon Simeonov 외 6인 공저, 2002
10. 전문 개발자를 위한 자바 웹 서비스, 피어슨에듀케이션코리아(PTG), 하비 디텔 외 5인 공저, 2003
11. Professional Apache Tomcat 5 ,WROX Press, Chanoch Wiggers 외 1인 공저, 2005
12. Professional Apache 2.0, WROX Press, Peter Wainwright, 2005
13. 모바일 디바이스에서 닷넷 애플리케이션 구축하기, 정보문화사, 앤디위글리, 피터 락스버그 공저/문봉재 역, 2004
14. 농업정보화, 농민신문사 조한근, 1997
15. 농산물 유통론, 전남대학교출판부, 전태갑, 2005
16. java.sun.com
17. www.mysql.org
18. tomcat.apache.org
19. httpd.apache.org
20. struts.apache.org

부 록 1

외부 서비스를 위한 WSDL

```
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions targetNamespace="http://userAuth.webservice" xmlns:a
pachesoap="http://xml.apache.org/xml-soap" xmlns:impl="http://userAuth.w
ebservice" xmlns:intf="http://userAuth.webservice" xmlns:soapenc="http://s
chemas.xmlsoap.org/soap/encoding/" xmlns:tns1="http://unit.acai.com" xmln
s:tns2="http://main.acai.com" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/
" xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:xsd="htt
p://www.w3.org/2001/XMLSchema">
- <!--
WSDL created by Apache Axis version: 1.4
Built on Dec 09, 2005 (03:14:07 GMT+00:00)

  -->
- <wsdl:types>
- <schema targetNamespace="http://main.acai.com" xmlns="http://www.w
3.org/2001/XMLSchema">
  <import namespace="http://userAuth.webservice" />
  <import namespace="http://unit.acai.com" />
  <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
- <complexType name="RegUser">
- <sequence>
  <element name="user" nillable="true" type="tns1:User" />
  <element name="myUser" nillable="true" type="tns1:User" />
  </sequence>
  </complexType>
```

```
  </schema>
- <schema targetNamespace="http://unit.acai.com" xmlns="http://www.w3.
org/2001/XMLSchema">
  <import namespace="http://userAuth.webservice" />
  <import namespace="http://main.acai.com" />
  <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
- <complexType name="User">
- <sequence>
  <element name="ID" nillable="true" type="xsd:string" />
  <element name="address" nillable="true" type="xsd:string" />
  <element name="address2" nillable="true" type="xsd:string" />
  <element name="addressCd" nillable="true" type="xsd:string" />
  <element name="confirm" type="xsd:boolean" />
  <element name="email" nillable="true" type="xsd:string" />
  <element name="grade" type="xsd:int" />
  <element name="mobile" nillable="true" type="xsd:string" />
  <element name="myRegUser" nillable="true" type="tns2:RegUser" />
  <element name="name" nillable="true" type="xsd:string" />
  <element name="password" nillable="true" type="xsd:string" />
  <element name="phone" nillable="true" type="xsd:string" />
  <element name="regNum" nillable="true" type="xsd:string" />
  </sequence>
  </complexType>
- <complexType name="UserE">
- <sequence>
  <element name="QCMNum" nillable="true" type="xsd:string" />
  <element name="account" nillable="true" type="xsd:string" />
  <element name="accountName" nillable="true" type="xsd:string" />
  <element name="bank" nillable="true" type="xsd:string" />
  <element name="bankNm" nillable="true" type="xsd:string" />
```

```
<element name="id" nillable="true" type="xsd:string" />
<element name="jakmokban" nillable="true" type="xsd:string" />
<element name="nonghup" nillable="true" type="xsd:string" />
<element name="nonghupNm" nillable="true" type="xsd:string" />
<element name="originCd" nillable="true" type="xsd:string" />
<element name="originNm" nillable="true" type="xsd:string" />
</sequence>
</complexType>
- <complexType name="ReceiptProduct">
- <sequence>
<element name="change" type="xsd:int" />
<element name="complete" type="xsd:int" />
<element name="gradeCode" nillable="true" type="xsd:string" />
<element name="gwasu" type="xsd:int" />
<element name="itemCode" nillable="true" type="xsd:string" />
<element name="quantity" type="xsd:int" />
<element name="recNum" type="xsd:int" />
<element name="shipDate" nillable="true" type="xsd:string" />
<element name="weight" type="xsd:int" />
</sequence>
</complexType>
- <complexType name="ReceiptAuction">
- <sequence>
<element name="ACCOUNT_NUM" nillable="true" type="xsd:string" />
<element name="CARGO_FEE" type="xsd:int" />
<element name="DEALER_ID" type="xsd:int" />
<element name="EXPENSE" type="xsd:int" />
<element name="LOW_COST" type="xsd:int" />
<element name="MANAGER_ID" type="xsd:int" />
<element name="MANAGE_NUM" nillable="true" type="xsd:string" />
```

```
<element name="RECEIPT" nillable="true" type="xsd:string" />
<element name="REC_NUM" type="xsd:int" />
<element name="REST" nillable="true" type="xsd:string" />
<element name="dealerName" nillable="true" type="xsd:string" />
<element name="managerName" nillable="true" type="xsd:string" />
<element name="receiptName" nillable="true" type="xsd:string" />
<element name="truckId" type="xsd:int" />
<element name="truckNum" nillable="true" type="xsd:string" />
</sequence>
</complexType>
- <complexType name="ReceiptMain">
- <sequence>
<element name="ID" nillable="true" type="xsd:string" />
<element name="change" type="xsd:int" />
<element name="complete" type="xsd:int" />
<element name="jakmokban" nillable="true" type="xsd:string" />
<element name="locationCode" type="xsd:int" />
<element name="nhName" nillable="true" type="xsd:string" />
<element name="nonghup" nillable="true" type="xsd:string" />
<element name="qcNum" type="xsd:int" />
<element name="recNum" type="xsd:int" />
<element name="shipDate" nillable="true" type="xsd:string" />
<element name="shipMethod" type="xsd:int" />
</sequence>
</complexType>
- <complexType name="RceiptProductServ">
- <sequence>
<element name="complete" type="xsd:int" />
<element name="grade" nillable="true" type="xsd:string" />
<element name="gradeCode" nillable="true" type="xsd:string" />
```

<element name="gwasu" type="xsd:int" />
<element name="item" nillable="true" type="xsd:string" />
<element name="itemCode" nillable="true" type="xsd:string" />
<element name="pummok" nillable="true" type="xsd:string" />
<element name="quantity" type="xsd:int" />
<element name="recNum" type="xsd:int" />
<element name="shipDate" nillable="true" type="xsd:string" />
<element name="weight" nillable="true" type="xsd:string" />
<element name="weightCode" type="xsd:int" />
</sequence>
</complexType>
- <complexType name="Receipt2">
- <sequence>
<element name="receiptAuction" nillable="true" type="tns1:ReceiptAuctio
n" />
<element name="receiptMain" nillable="true" type="tns1:ReceiptMain" />
<element name="receiptProductList" nillable="true" type="impl:ArrayOf_t
ns2_RceiptProductServ" />
</sequence>
</complexType>
- <complexType name="CommManage">
- <sequence>
<element name="name" nillable="true" type="xsd:string" />
<element name="no" type="xsd:int" />
<element name="use" type="xsd:boolean" />
</sequence>
</complexType>
- <complexType name="Grade">
- <sequence>
<element name="code" nillable="true" type="xsd:string" />

<element name="name" nillable="true" type="xsd:string" />
</sequence>
</complexType>
</schema>
- <schema targetNamespace="http://userAuth.webservice" xmlns="http://www.w3.org/2001/XMLSchema">
<import namespace="http://unit.acai.com" />
<import namespace="http://main.acai.com" />
<import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
- <complexType name="ArrayOf_soapenc_string">
- <complexContent>
- <restriction base="soapenc:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]" />
</restriction>
</complexContent>
</complexType>
- <complexType name="ArrayOf_tns2_ReceiptProduct">
- <complexContent>
- <restriction base="soapenc:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="tns1:ReceiptProduct[]" />
</restriction>
</complexContent>
</complexType>
- <complexType name="ArrayOf_tns2_RceiptProductServ">
- <complexContent>
- <restriction base="soapenc:Array">
<attribute ref="soapenc:arrayType" wsdl:arrayType="tns1:RceiptProductServ[]" />
</restriction>

```
  </complexContent>
  </complexType>
- <complexType name="ArrayOf_tns2_CommManage">
- <complexContent>
- <restriction base="soapenc:Array">
  <attribute ref="soapenc:arrayType" wsdl:arrayType="tns1:CommManage
[]" />
  </restriction>
  </complexContent>
  </complexType>
- <complexType name="ArrayOf_tns2_Grade">
- <complexContent>
- <restriction base="soapenc:Array">
  <attribute ref="soapenc:arrayType" wsdl:arrayType="tns1:Grade[]" />
  </restriction>
  </complexContent>
  </complexType>
  </schema>
  </wsdl:types>
- <wsdl:message name="getUserEResponse">
  <wsdl:part name="getUserEReturn" type="tns1:UserE" />
  </wsdl:message>
- <wsdl:message name="login1Response">
  <wsdl:part name="login1Return" type="xsd:int" />
  </wsdl:message>
  <wsdl:message name="getMainCodeRequest" />
- <wsdl:message name="findReceiptResponse">
  <wsdl:part name="findReceiptReturn" type="tns1:Receipt2" />
  </wsdl:message>
- <wsdl:message name="calCargoFeeRequest">
```

<wsdl:part name="in0" type="xsd:int" />
<wsdl:part name="in1" type="xsd:int" />
</wsdl:message>
<wsdl:message name="getManagerRequest" />
- <wsdl:message name="getRegNumRequest">
<wsdl:part name="in0" type="soapenc:string" />
</wsdl:message>
- <wsdl:message name="getDealerResponse">
<wsdl:part name="getDealerReturn" type="impl:ArrayOf_tns2_CommMan
age" />
</wsdl:message>
- <wsdl:message name="confirmCompleteRequest">
<wsdl:part name="in0" type="xsd:int" />
<wsdl:part name="in1" type="soapenc:string" />
<wsdl:part name="in2" type="tns1:ReceiptAuction" />
</wsdl:message>
- <wsdl:message name="getItemListRequest">
<wsdl:part name="in0" type="soapenc:string" />
</wsdl:message>
- <wsdl:message name="getUserERequest">
<wsdl:part name="in0" type="soapenc:string" />
</wsdl:message>
- <wsdl:message name="getManagerResponse">
<wsdl:part name="getManagerReturn" type="impl:ArrayOf_tns2_CommM
anage" />
</wsdl:message>
<wsdl:message name="getTruckRequest" />
- <wsdl:message name="login1Request">
<wsdl:part name="in0" type="soapenc:string" />
<wsdl:part name="in1" type="soapenc:string" />

```
  </wsdl:message>
- <wsdl:message name="getTruckResponse">
  <wsdl:part name="getTruckReturn" type="impl:ArrayOf_tns2_CommMana
ge" />
  </wsdl:message>
- <wsdl:message name="getCargoFeeRequest">
  <wsdl:part name="in0" type="xsd:int" />
  </wsdl:message>
- <wsdl:message name="createRequest">
  <wsdl:part name="in0" type="xsd:int" />
  <wsdl:part name="in1" type="soapenc:string" />
  <wsdl:part name="in2" type="tns1:ReceiptAuction" />
  <wsdl:part name="in3" type="impl:ArrayOf_tns2_RceiptProductServ" />
  </wsdl:message>
- <wsdl:message name="getUserRequest">
  <wsdl:part name="in0" type="soapenc:string" />
  </wsdl:message>
  <wsdl:message name="getDealerRequest" />
- <wsdl:message name="updateRequest">
  <wsdl:part name="in0" type="xsd:int" />
  <wsdl:part name="in1" type="tns1:ReceiptAuction" />
  <wsdl:part name="in2" type="impl:ArrayOf_tns2_RceiptProductServ" />
  </wsdl:message>
- <wsdl:message name="updateProductRequest">
  <wsdl:part name="in0" type="xsd:int" />
  <wsdl:part name="in1" type="impl:ArrayOf_tns2_RceiptProductServ" />
  <wsdl:part name="in2" type="soapenc:string" />
  </wsdl:message>
  <wsdl:message name="confirmCompleteResponse" />
- <wsdl:message name="getRegNumResponse">
```

<wsdl:part name="getRegNumReturn" type="impl:ArrayOf_soapenc_string" />
</wsdl:message>
<wsdl:message name="createResponse" />
- <wsdl:message name="getUserResponse">
<wsdl:part name="getUserReturn" type="tns1:User" />
</wsdl:message>
- <wsdl:message name="getItemListResponse">
<wsdl:part name="getItemListReturn" type="impl:ArrayOf_tns2_ReceiptProduct" />
</wsdl:message>
- <wsdl:message name="calCargoFeeResponse">
<wsdl:part name="calCargoFeeReturn" type="xsd:int" />
</wsdl:message>
<wsdl:message name="updateResponse" />
- <wsdl:message name="getMainCodeResponse">
<wsdl:part name="getMainCodeReturn" type="impl:ArrayOf_tns2_Grade" />
</wsdl:message>
<wsdl:message name="updateProductResponse" />
- <wsdl:message name="getCargoFeeResponse">
<wsdl:part name="getCargoFeeReturn" type="xsd:int" />
</wsdl:message>
- <wsdl:message name="findReceiptRequest">
<wsdl:part name="in0" type="xsd:int" />
</wsdl:message>
- <wsdl:portType name="UserAuthIF">
- <wsdl:operation name="login1" parameterOrder="in0 in1">
<wsdl:input message="impl:login1Request" name="login1Request" />
<wsdl:output message="impl:login1Response" name="login1Response" />

```
  </wsdl:operation>
- <wsdl:operation name="getUser" parameterOrder="in0">
  <wsdl:input message="impl:getUserRequest" name="getUserRequest" />
  <wsdl:output message="impl:getUserResponse" name="getUserResponse"
/>
  </wsdl:operation>
- <wsdl:operation name="getUserE" parameterOrder="in0">
  <wsdl:input message="impl:getUserERequest" name="getUserERequest"
/>
  <wsdl:output message="impl:getUserEResponse" name="getUserERespon
se" />
  </wsdl:operation>
- <wsdl:operation name="getRegNum" parameterOrder="in0">
  <wsdl:input message="impl:getRegNumRequest" name="getRegNumRequ
est" />
  <wsdl:output message="impl:getRegNumResponse" name="getRegNumRe
sponse" />
  </wsdl:operation>
- <wsdl:operation name="getItemList" parameterOrder="in0">
  <wsdl:input message="impl:getItemListRequest" name="getItemListReque
st" />
  <wsdl:output message="impl:getItemListResponse" name="getItemListRes
ponse" />
  </wsdl:operation>
- <wsdl:operation name="findReceipt" parameterOrder="in0">
  <wsdl:input message="impl:findReceiptRequest" name="findReceiptReques
t" />
  <wsdl:output message="impl:findReceiptResponse" name="findReceiptRes
ponse" />
  </wsdl:operation>
```

- <wsdl:operation name="getDealer">
<wsdl:input message="impl:getDealerRequest" name="getDealerRequest" />
<wsdl:output message="impl:getDealerResponse" name="getDealerResponse" />
</wsdl:operation>
- <wsdl:operation name="getTruck">
<wsdl:input message="impl:getTruckRequest" name="getTruckRequest" />
<wsdl:output message="impl:getTruckResponse" name="getTruckResponse" />
</wsdl:operation>
- <wsdl:operation name="getMainCode">
<wsdl:input message="impl:getMainCodeRequest" name="getMainCodeRequest" />
<wsdl:output message="impl:getMainCodeResponse" name="getMainCodeResponse" />
</wsdl:operation>
- <wsdl:operation name="updateProduct" parameterOrder="in0 in1 in2">
<wsdl:input message="impl:updateProductRequest" name="updateProductRequest" />
<wsdl:output message="impl:updateProductResponse" name="updateProductResponse" />
</wsdl:operation>
- <wsdl:operation name="confirmComplete" parameterOrder="in0 in1 in2">
<wsdl:input message="impl:confirmCompleteRequest" name="confirmCompleteRequest" />
<wsdl:output message="impl:confirmCompleteResponse" name="confirmCompleteResponse" />
</wsdl:operation>

<wsdl:operation name="getCargoFee" parameterOrder="in0">
<wsdl:input message="impl:getCargoFeeRequest" name="getCargoFeeRequest" />
<wsdl:output message="impl:getCargoFeeResponse" name="getCargoFeeResponse" />
</wsdl:operation>
<wsdl:operation name="calCargoFee" parameterOrder="in0 in1">
<wsdl:input message="impl:calCargoFeeRequest" name="calCargoFeeRequest" />
<wsdl:output message="impl:calCargoFeeResponse" name="calCargoFeeResponse" />
</wsdl:operation>
<wsdl:operation name="create" parameterOrder="in0 in1 in2 in3">
<wsdl:input message="impl:createRequest" name="createRequest" />
<wsdl:output message="impl:createResponse" name="createResponse" />
</wsdl:operation>
<wsdl:operation name="update" parameterOrder="in0 in1 in2">
<wsdl:input message="impl:updateRequest" name="updateRequest" />
<wsdl:output message="impl:updateResponse" name="updateResponse" />
</wsdl:operation>
<wsdl:operation name="getManager">
<wsdl:input message="impl:getManagerRequest" name="getManagerRequest" />
<wsdl:output message="impl:getManagerResponse" name="getManagerResponse" />
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="userAuthSoapBinding" type="impl:UserAuthIF">
<wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soa

p/http" />
- <wsdl:operation name="login1">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="login1Request">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="login1Response">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="getUser">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="getUserRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="getUserResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="getUserE">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="getUserERequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="getUserEResponse">

<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="getRegNum">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="getRegNumRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="getRegNumResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="getItemList">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="getItemListRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="getItemListResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="findReceipt">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="findReceiptRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin

g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="findReceiptResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="getDealer">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="getDealerRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="getDealerResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="getTruck">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="getTruckRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="getTruckResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="getMainCode">

<wsdlsoap:operation soapAction="" />
- <wsdl:input name="getMainCodeRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="getMainCodeResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="updateProduct">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="updateProductRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="updateProductResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
- <wsdl:operation name="confirmComplete">
<wsdlsoap:operation soapAction="" />
- <wsdl:input name="confirmCompleteRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
- <wsdl:output name="confirmCompleteResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />

```
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="getCargoFee">
  <wsdlsoap:operation soapAction="" />
- <wsdl:input name="getCargoFeeRequest">
  <wsdlsoap:body  encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
  </wsdl:input>
- <wsdl:output name="getCargoFeeResponse">
  <wsdlsoap:body  encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="calCargoFee">
  <wsdlsoap:operation soapAction="" />
- <wsdl:input name="calCargoFeeRequest">
  <wsdlsoap:body  encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
  </wsdl:input>
- <wsdl:output name="calCargoFeeResponse">
  <wsdlsoap:body  encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="create">
  <wsdlsoap:operation soapAction="" />
- <wsdl:input name="createRequest">
  <wsdlsoap:body  encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
  </wsdl:input>
```

<wsdl:output name="createResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="update">
<wsdlsoap:operation soapAction="" />
<wsdl:input name="updateRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
<wsdl:output name="updateResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="getManager">
<wsdlsoap:operation soapAction="" />
<wsdl:input name="getManagerRequest">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:input>
<wsdl:output name="getManagerResponse">
<wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encodin
g/" namespace="http://userAuth.webservice" use="encoded" />
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="UserAuthIFService">
<wsdl:port binding="impl:userAuthSoapBinding" name="userAuth">

```
    <wsdlsoap:address    location="http://220.67.220.162:8080/regManage/service
s/userAuth" />
    </wsdl:port>
    </wsdl:service>
    </wsdl:definitions>
```

부 록 2

WSDL을 이용한 클래스 구현

```
namespace SmartDeviceApplication2.WebService {
    using System.Diagnostics;
    using System.Xml.Serialization;
    using System;
    using System.Web.Services.Protocols;
    using System.ComponentModel;
    using System.Web.Services;

    /// <remarks/>
    [System.Diagnostics.DebuggerStepThroughAttribute()]
    [System.ComponentModel.DesignerCategoryAttribute("code")]
    [System.Web.Services.WebServiceBindingAttribute(Name="userAuthSo
apBinding", Namespace="http://userAuth.webservice")]
    [System.Xml.Serialization.SoapIncludeAttribute(typeof(Grade))]
    [System.Xml.Serialization.SoapIncludeAttribute(typeof(CommManage))]
    [System.Xml.Serialization.SoapIncludeAttribute(typeof(RceiptProductSer
v))]
    [System.Xml.Serialization.SoapIncludeAttribute(typeof(ReceiptProduct))]
    public class UserAuthIFService : System.Web.Services.Protocols.Soap
HttpClientProtocol {

        /// <remarks/>
        public UserAuthIFService() {
            this.Url = "http://220.67.220.162:8080/regManage/services/user
Auth";
```

```
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("login1Return")]
        public int login1(string in0, string in1) {
            object[] results = this.Invoke("login1", new object[] {
                        in0,
                        in1});
            return ((int)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult Beginlogin1(string in0, string in1, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("login1", new object[] {
                        in0,
                        in1}, callback, asyncState);
        }

        /// <remarks/>
        public int Endlogin1(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((int)(results[0]));
        }

        /// <remarks/>
```

```
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getUserRe
turn")]
        public User getUser(string in0) {
            object[] results = this.Invoke("getUser", new object[] {
                        in0});
            return ((User)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetUser(string in0, System.Asyn
cCallback callback, object asyncState) {
            return this.BeginInvoke("getUser", new object[] {
                        in0}, callback, asyncState);
        }

        /// <remarks/>
        public User EndgetUser(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((User)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getUserER
eturn")]
```

```
        public UserE getUserE(string in0) {
            object[] results = this.Invoke("getUserE", new object[] {
                        in0});
            return ((UserE)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetUserE(string in0, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("getUserE", new object[] {
                        in0}, callback, asyncState);
        }

        /// <remarks/>
        public UserE EndgetUserE(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((UserE)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getRegNumReturn")]
        public string[] getRegNum(string in0) {
            object[] results = this.Invoke("getRegNum", new object[] {
                        in0});
            return ((string[])(results[0]));
        }
```

```
        /// <remarks/>
        public System.IAsyncResult BegingetRegNum(string in0, System.
AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("getRegNum", new object[] {
                        in0}, callback, asyncState);
        }

        /// <remarks/>
        public string[] EndgetRegNum(System.IAsyncResult asyncResult)
{
            object[] results = this.EndInvoke(asyncResult);
            return ((string[])(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getItemLis
tReturn")]
        public ReceiptProduct[] getItemList(string in0) {
            object[] results = this.Invoke("getItemList", new object[] {
                        in0});
            return ((ReceiptProduct[])(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetItemList(string in0, System.A
syncCallback callback, object asyncState) {
```

```
            return this.BeginInvoke("getItemList", new object[] {
                        in0}, callback, asyncState);
        }

        /// <remarks/>
        public ReceiptProduct[] EndgetItemList(System.IAsyncResult asyn
cResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((ReceiptProduct[])(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("findReceip
tReturn")]
        public Receipt2 findReceipt(int in0) {
            object[] results = this.Invoke("findReceipt", new object[] {
                        in0});
            return ((Receipt2)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginfindReceipt(int in0, System.Asyn
cCallback callback, object asyncState) {
            return this.BeginInvoke("findReceipt", new object[] {
                        in0}, callback, asyncState);
        }
```

```
        /// <remarks/>
        public Receipt2 EndfindReceipt(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((Receipt2)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getDealerReturn")]
        public CommManage[] getDealer() {
            object[] results = this.Invoke("getDealer", new object[0]);
            return ((CommManage[])(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetDealer(System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("getDealer", new object[0], callback, asyncState);
        }

        /// <remarks/>
        public CommManage[] EndgetDealer(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((CommManage[])(results[0]));
```

```
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getTruckReturn")]
        public CommManage[] getTruck() {
            object[] results = this.Invoke("getTruck", new object[0]);
            return ((CommManage[])(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetTruck(System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("getTruck", new object[0], callback, asyncState);
        }

        /// <remarks/>
        public CommManage[] EndgetTruck(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((CommManage[])(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://use
```

```
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getMainCo
deReturn")]
        public Grade[] getMainCode() {
            object[] results = this.Invoke("getMainCode", new object[0]);
            return ((Grade[])(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetMainCode(System.AsyncCallb
ack callback, object asyncState) {
            return this.BeginInvoke("getMainCode", new object[0], callbac
k, asyncState);
        }

        /// <remarks/>
        public Grade[] EndgetMainCode(System.IAsyncResult asyncResult)
{
            object[] results = this.EndInvoke(asyncResult);
            return ((Grade[])(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        public void updateProduct(int in0, RceiptProductServ[] in1, string
in2) {
            this.Invoke("updateProduct", new object[] {
                        in0,
```

```
                        in1,
                        in2});
        }

        /// <remarks/>
        public System.IAsyncResult BeginupdateProduct(int in0, RceiptProductServ[] in1, string in2, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("updateProduct", new object[] {
                        in0,
                        in1,
                        in2}, callback, asyncState);
        }

        /// <remarks/>
        public void EndupdateProduct(System.IAsyncResult asyncResult) {
            this.EndInvoke(asyncResult);
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        public void confirmComplete(int in0, string in1, ReceiptAuction in2) {
            this.Invoke("confirmComplete", new object[] {
                        in0,
                        in1,
                        in2});
        }
```

```
        /// <remarks/>
        public System.IAsyncResult BeginconfirmComplete(int in0, string i
n1, ReceiptAuction in2, System.AsyncCallback callback, object asyncState)
{
            return this.BeginInvoke("confirmComplete", new object[] {
                        in0,
                        in1,
                        in2}, callback, asyncState);
        }

        /// <remarks/>
        public void EndconfirmComplete(System.IAsyncResult asyncResul
t) {
            this.EndInvoke(asyncResult);
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getCargoF
eeReturn")]
        public int getCargoFee(int in0) {
            object[] results = this.Invoke("getCargoFee", new object[] {
                        in0});
            return ((int)(results[0]));
        }

        /// <remarks/>
```

```
        public System.IAsyncResult BegingetCargoFee(int in0, System.As
yncCallback callback, object asyncState) {
            return this.BeginInvoke("getCargoFee", new object[] {
                        in0}, callback, asyncState);
        }

        /// <remarks/>
        public int EndgetCargoFee(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((int)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", Reque
stNamespace="http://userAuth.webservice", ResponseNamespace="http://use
rAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("calCargoF
eeReturn")]
        public int calCargoFee(int in0, int in1) {
            object[] results = this.Invoke("calCargoFee", new object[] {
                        in0,
                        in1});
            return ((int)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegincalCargoFee(int in0, int in1, Sys
tem.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("calCargoFee", new object[] {
                        in0,
```

```
                        in1}, callback, asyncState);
        }

        /// <remarks/>
        public int EndcalCargoFee(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((int)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        public void create(int in0, string in1, ReceiptAuction in2, RceiptProductServ[] in3) {
            this.Invoke("create", new object[] {
                        in0,
                        in1,
                        in2,
                        in3});
        }

        /// <remarks/>
        public System.IAsyncResult Begincreate(int in0, string in1, ReceiptAuction in2, RceiptProductServ[] in3, System.AsyncCallback callback, object asyncState) {
    return this.BeginInvoke("create", new object[] {
                        in0,
                        in1,
                        in2,
```

```
                    in3}, callback, asyncState);
        }

        /// <remarks/>
        public void Endcreate(System.IAsyncResult asyncResult) {
            this.EndInvoke(asyncResult);
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        public void update(int in0, ReceiptAuction in1, RceiptProductServ[] in2) {
            this.Invoke("update", new object[] {
                    in0,
                    in1,
                    in2});
        }

        /// <remarks/>
        public System.IAsyncResult Beginupdate(int in0, ReceiptAuction in1, RceiptProductServ[] in2, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("update", new object[] {
                    in0,
                    in1,
                    in2}, callback, asyncState);
        }
```

```
        /// <remarks/>
        public void Endupdate(System.IAsyncResult asyncResult) {
            this.EndInvoke(asyncResult);
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapRpcMethodAttribute("", RequestNamespace="http://userAuth.webservice", ResponseNamespace="http://userAuth.webservice")]
        [return: System.Xml.Serialization.SoapElementAttribute("getManagerReturn")]
        public CommManage[] getManager() {
            object[] results = this.Invoke("getManager", new object[0]);
            return ((CommManage[])(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BegingetManager(System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("getManager", new object[0], callback, asyncState);
        }

        /// <remarks/>
        public CommManage[] EndgetManager(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((CommManage[])(results[0]));
        }
    }
```

```
    /// <remarks/>
    [System.Xml.Serialization.SoapTypeAttribute("User", "http://unit.acai.com")]
    public class User {

        /// <remarks/>
        public string ID;

        /// <remarks/>
        public string address;

        /// <remarks/>
        public string address2;

        /// <remarks/>
        public string addressCd;

        /// <remarks/>
        public bool confirm;

        /// <remarks/>
        public string email;

        /// <remarks/>
        public int grade;

        /// <remarks/>
        public string mobile;
```

```
        /// <remarks/>
        public RegUser myRegUser;

        /// <remarks/>
        public string name;

        /// <remarks/>
        public string password;

        /// <remarks/>
        public string phone;

        /// <remarks/>
        public string regNum;
    }

    /// <remarks/>
    [System.Xml.Serialization.SoapTypeAttribute("RegUser", "http://main.ac
ai.com")]
    public class RegUser {

        /// <remarks/>
        public User user;

        /// <remarks/>
        public User myUser;
    }

    /// <remarks/>
    [System.Xml.Serialization.SoapTypeAttribute("Grade", "http://unit.acai.c
```

```
om")]
    public class Grade {

        /// <remarks/>
        public string code;

        /// <remarks/>
        public string name;
    }

    /// <remarks/>
    [System.Xml.Serialization.SoapTypeAttribute("CommManage", "http://unit.acai.com")]
    public class CommManage {

        /// <remarks/>
        public string name;

        /// <remarks/>
        public int no;

        /// <remarks/>
        public bool use;
    }

    /// <remarks/>
    [System.Xml.Serialization.SoapTypeAttribute("RceiptProductServ", "http://unit.acai.com")]
    public class RceiptProductServ {
```

```
        /// <remarks/>
        public int complete;

        /// <remarks/>
        public string grade;

        /// <remarks/>
        public string gradeCode;

        /// <remarks/>
        public int gwasu;

        /// <remarks/>
        public string item;

        /// <remarks/>
        public string itemCode;

        /// <remarks/>
        public string pummok;

        /// <remarks/>
        public int quantity;

        /// <remarks/>
        public int recNum;

        /// <remarks/>
        public string shipDate;
```

```
            /// <remarks/>
            public string weight;

            /// <remarks/>
            public int weightCode;
        }

        /// <remarks/>
        [System.Xml.Serialization.SoapTypeAttribute("ReceiptMain", "http://unit.acai.com")]
        public class ReceiptMain {

            /// <remarks/>
            public string ID;

            /// <remarks/>
            public int change;

            /// <remarks/>
            public int complete;

            /// <remarks/>
            public string jakmokban;

            /// <remarks/>
            public int locationCode;

            /// <remarks/>
            public string nhName;
```

```
            /// <remarks/>
            public string nonghup;

            /// <remarks/>
            public int qcNum;

            /// <remarks/>
            public int recNum;

            /// <remarks/>
            public string shipDate;

            /// <remarks/>
            public int shipMethod;
        }

        /// <remarks/>
        [System.Xml.Serialization.SoapTypeAttribute("ReceiptAuction", "http://unit.acai.com")]
        public class ReceiptAuction {

            /// <remarks/>
            public string ACCOUNT_NUM;

            /// <remarks/>
            public int CARGO_FEE;

            /// <remarks/>
            public int DEALER_ID;
```

```
/// <remarks/>
public int EXPENSE;

/// <remarks/>
public int LOW_COST;

/// <remarks/>
public int MANAGER_ID;

/// <remarks/>
public string MANAGE_NUM;

/// <remarks/>
public string RECEIPT;

/// <remarks/>
public int REC_NUM;

/// <remarks/>
public string REST;

/// <remarks/>
public string dealerName;

/// <remarks/>
public string managerName;

/// <remarks/>
public string receiptName;
```

```
            /// <remarks/>
            public int truckId;

            /// <remarks/>
            public string truckNum;
        }

        /// <remarks/>
        [System.Xml.Serialization.SoapTypeAttribute("Receipt2", "http://unit.acai.com")]
        public class Receipt2 {

            /// <remarks/>
            public ReceiptAuction receiptAuction;

            /// <remarks/>
            public ReceiptMain receiptMain;

            /// <remarks/>
            public RceiptProductServ[] receiptProductList;
        }

        /// <remarks/>
        [System.Xml.Serialization.SoapTypeAttribute("ReceiptProduct", "http://unit.acai.com")]
        public class ReceiptProduct {

            /// <remarks/>
            public int change;
```

```
        /// <remarks/>
        public int complete;

        /// <remarks/>
        public string gradeCode;

        /// <remarks/>
        public int gwasu;

        /// <remarks/>
        public string itemCode;

        /// <remarks/>
        public int quantity;

        /// <remarks/>
        public int recNum;

        /// <remarks/>
        public string shipDate;

        /// <remarks/>
        public int weight;
    }

    /// <remarks/>
    [System.Xml.Serialization.SoapTypeAttribute("UserE", "http://unit.acai.com")]
    public class UserE {
```

```
/// <remarks/>
public string QCMNum;

/// <remarks/>
public string account;

/// <remarks/>
public string accountName;

/// <remarks/>
public string bank;

/// <remarks/>
public string bankNm;

/// <remarks/>
public string id;

/// <remarks/>
public string jakmokban;

/// <remarks/>
public string nonghup;

/// <remarks/>
public string nonghupNm;

/// <remarks/>
public string originCd;
```

```
            /// <remarks/>
            public string originNm;
        }
    }
```