Unit Testing Plan

for POS System

- **Test Plan**
- **Test Design Specification**
- **Test Cases Specification**

Project Team

Team 3

Date

2017-11-06

Team Information

장혁준 허윤아 현인수 전상우

Table of Contents

1 Introduction

1.1 Objectives

본 문서는 건국대학교 소프트웨어공학개론 2017 학년도 2 학기 수업 팀 3 에서 POS System 의 Unit Test 를 수행하기 위해 작성한 Unit Testing Plan 이다. Test Case 는 SRA(Software Requirement Analysis)에 의거 하여 작성된다.

1.2 Background

POS System 이란 판매와 관련한 데이터를 일괄적으로 관리하고, 고객 정보를 수집하여 부가가치를 부가 가치를 향상시키는 시스템이다.

Unit Test 란 시스템의 최소 단위 모듈들이 각각 의도된대로 작동이 되는지 검증하는 것이다.

1.3 Scope

본 문서는 POS 시스템의 Unit Test 를 수행하기 위한 모든 것을 포함한다. POS 기능을 수행하기 위한 자원과 절차, 그 외 필요로 하는 환경 및 도구 등을 정의한다. Unit Test 에서 실시되는 모든 테스트는 최소 단위의 모듈들을 대상으로 하며, 구현된 모듈이 요구사항을 만족하는지를 테스트한다.

1.4 Project plan

POS 시스템의 SRA, SDA 등 명세서(Specification)을 기반으로 Unit test 를 실시한다. Unit test 는 Ctest tool 을 사용하고, Cygwin 혹은 Linux 환경에서 이루어진다. Program Source code 는 일정 주기마다 팀원들과 build 및 Unit Test 를 수행한다.

1.5 Configuration management plan

POS 시스템은 팀원들과 일정 주기마다 Build 및 Unit test 를 통해 수정, 관리된다. 변경 및 수정사항은 지속적으로 통합되고 Test 된다.

1.6 References

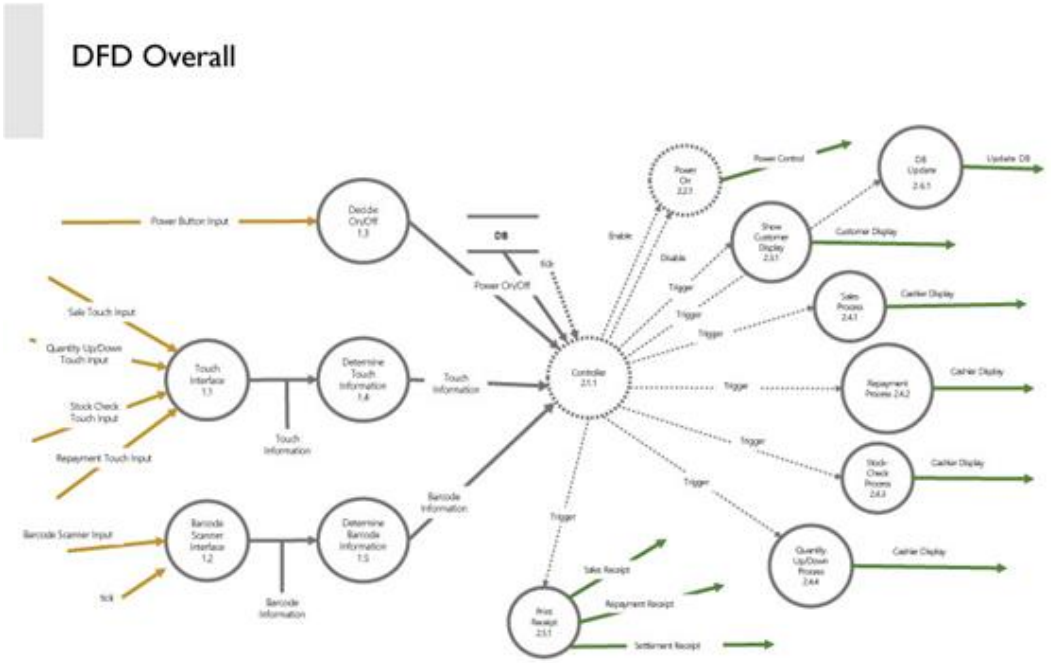
T3-2017.POS.SRA- 2.1

T3-2017.POS.SDS- 2.1

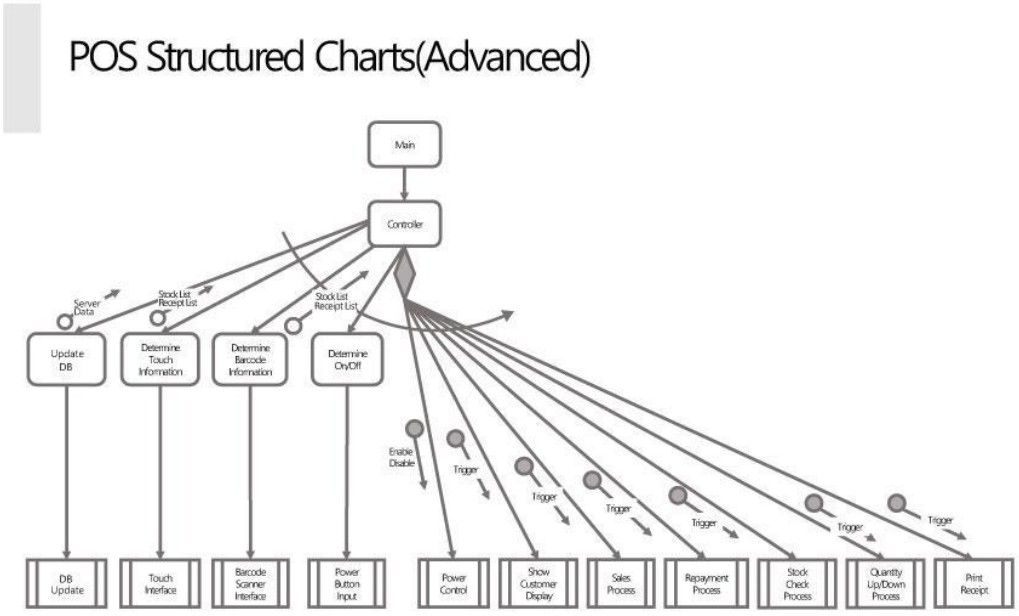
2 Test items

SASD 절차를 통해 디자인 한 POS 시스템을 기반으로 test item 을 구성하고, 일정 수준 이상 구현, test 한다. input 과 output 을 처리하는 함수가 SRS 의 요구사항에 맞춰 정상적으로 작동되는지를 test 한다.

1) DFD Overall

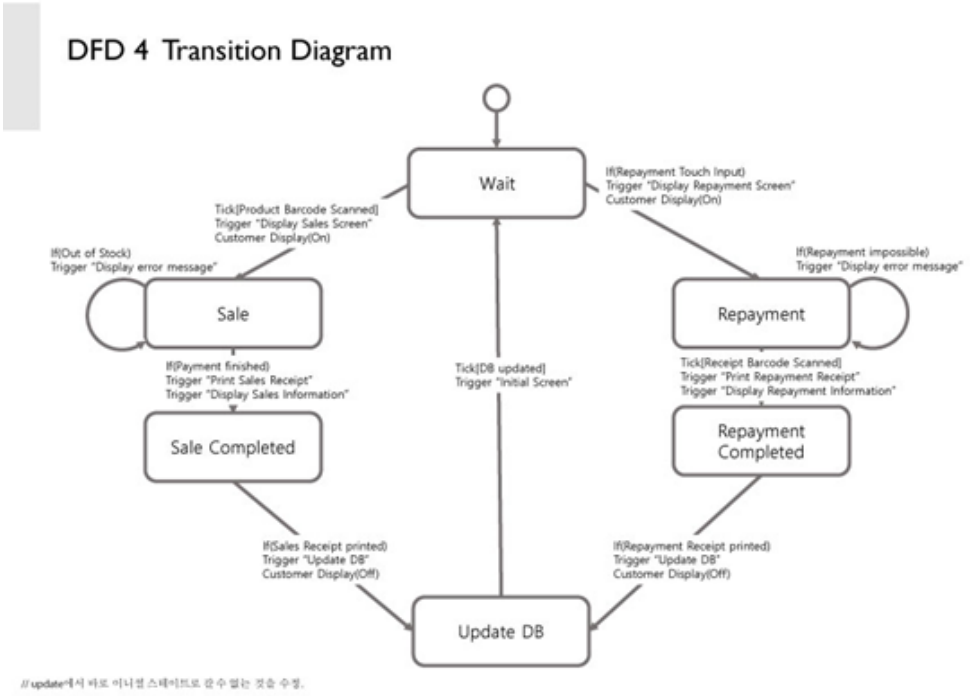


2) Structured Charts



// Update db의 표현이 되어있지 않았으나 표현.

3) Transition Diagram



3 Features to be tested

(1) Process in SRA : 각 프로세스가 가지고 있는 요구사항을 만족하는 지를 Test 한다.

(2) Modules in SDS : 각 모듈이 가지고 있는 데이터 인터페이스를 Test 한다

<Table 1>

Unit Test 는 함수가 원활하게 작동하는지를 검사하는 테스트이므로 Control Process 에 한해 Unit Test 를 진행한다.

ID	Name	Description
2.1.1	Controller	DB data, Power data, Touch data, Barcode data, Clock data 를 받아들여 명령을 내린다.
2.2.1	Power On	Power On/Off. Power =1 일 때 POS 가 시작된다.
2.4.1	Sales Process	판매상황에서 결제완료 후 디스플레이를 출력 한다. 예시) total : 20000 cash : 30000 change : 10000

		환불상황에서 환불완료 후 디스플레이를 출력 한다. 예시) date : YYMMDDhhmm refund total : 20000
2.4.1	Sales Process	판매 버튼을 눌렀을 때 판매 상황으로 넘어간다.
2.4.2	Repayment Process	환불 버튼을 눌렀을 때 환불 상황으로 넘어간다.
2.4.3	Stock-Check Process	재고 확인 버튼을 눌렀을 때 재고 확인 상황으로 넘어간다.
2.4.4	Quantity Up/Down Process	바코드를 찍은 물품에 수량을 추가하거나 줄인다.
2.6.1	DB Update	판매상황에서 결제 완료후 재고서버의 상품 개수를 업데이트한다. (product.txt) 환불상황에서 환불 완료후.재고서버의 상품 개수를 업데이트한다.(product.txt) 판매상황에서 재고 서버의 판매 정보를 추가한다.(management.txt)
2.5.1	Print Receipt	판매상황에서 결제완료 후 판매영수증을 출력한다. (sale_YYYYMMDDhhmm.txt) 환불상황에서 환불완료 후 환불영수증을 출력한다. (refund_YYYYMMDDhhmm.txt) 재고확인 상황에서 재고확인을 출력한다. (stock_YYYYMMDD.txt)
2.3.1	Show Customer Display	판매상황에서 결제완료 후 디스플레이를 출력 한다. 예시) total : 20000 cash : 30000 change : 10000 List : 상품리스트... 환불상황에서 환불완료 후 디스플레이를 출력 한다. 예시) date : YYMMDDhhmm

		refund total : 20000 List : 환불 상품리스트...

4 Features not to be tested

(1) Process in SRA : 단순 데이터 전달 프로세스 및 interface 의 작동은 test 대상에서 제외한다.

(2) Modules in SDS : 테스트하지 않을 Process 리스트(<Table 2> 참조)

(3) 테스트 할 수 있을 정도까지 구현하지 못한 부분(일부)

<Table 2>

ID	Name	Description
1.1	Touch Interface	화면에서 각 버튼들을 입력 받는다.
1.2	Barcode Scanner Interface	바코드 input 을 입력 받는다.
1.3	Decide On/off	파워 버튼 input 을 전달한다.
1.4	Determine Touch information	버튼 입력들을 전달한다
1.5	Determine Barcode information	바코드의 입력들을 전달한다

5 Approach

POS 시스템의 Program Source Code 및 Unit Test 를 위한 Test Code 는 Cygwin(gcc) 환경에서 이루어지며 Program Source Code 의 변경 및 수정사항은 지속적으로 통합되고 테스트된다

6 Item pass/fail criteria

Functional Test pass/fail criteria : 각 모듈은 요구사항을 모두 만족하여야 한다.

7 Unit test design specification

7.1 Test design specification identifier

POS_[Category]_[number].unit

7.2 Features to be tested

<Table 1> 참조

7.3 Approach refinements

각 모듈이 요구사항을 만족하는지 확인하기 위해, 요구사항에 정의된 내용에 기반하여 test case 를 작성한다. 그 이외의 사항에 대해서는 test case 를 작성하지 않는다.

7.4 Test identification

<Table 3>

Identifier	Feature(Process DFD)	Valid/Invalid(value)
POS_Controller_Customer_01	2.3.1 Customer Display	고객 판매 화면
POS_Controller_Customer_02	2.3.1 Customer Display	고객 환불 화면
POS_CashierDisplay_01	2.7.1 Cashier Display	캐셔 판매 화면 productNow productList[] (구조체 배열) typedef struct productNow { char name[15]; int price; int quantity; } productNow;
POS_CashierDisplay_02	2.7.1 Cashier Display	캐셔 환불 화면 productR list[],int barcode (구조체배열 ,바코드정보) typedef struct productR { char name[15]; int price; int quantity; int salePrice; } productR;

POS_Controller_Sales_01	2.4.1 Sale Process	menu==1
POS_Controller_Repayment_01	2.4.2 Repayment Process	menu==2
POS_Controller_Stock_01	2.4.3 Stock Check Process	menu==3
POS_Controller_Quantity_01	2.4.4 Quantity Up/Down Process	NOTIMPLEMENTED
POS_Controller_DB_01	2.6.1 DB Update	int s[7] (물품별 판매/환불 갯수들을 담고 있음)
POS_Power_01	2.2.1 Power On	전원 켜기 power==1
POS_Customer_01	2.3.1 Show Customer Display	고객 화면 출력(판매) productNow productList[] (구조체 배열) typedef struct productNow { char name[15]; int price; int quantity; } productNow;
POS_Customer_02	2.3.1 Show Customer Display	고객 화면 출력(환불) productR list[],int barcode (구조체배열 ,바코드정보) typedef struct productR { char name[15]; int price;

		int quantity; int salePrice; } productR;
POS_Sales_01	2.4.1 Sales Process	상품 입력 char barcodeData[5] barcodeData == 001/010/011/100/101/110/111
POS_Sales_02	2.4.1 Sales Process	상품 결제 Int cash 입력한 금액이 부족할 경우 Printf("Not Enough Money")
POS_Repayment_01	2.5.1 Repayment Process	환불 영수증 입력 int barcode
POS_Repayment_02	2.5.1 Repayment Process	바코드유효성 확인 int checkBracodeValid(barcode) ==1
POS_PrintReceipt_01	2.5.3.1 Print Receipt_SalesReceipt	판매 영수증 출력 productNow productList[] (구조체 배열) typedef struct productNow { char name[15]; int price; int quantity;

		} productNow;
POS_PrintReceipt_02	2.5.3.2 Print Receipt_RepaymentReceipt	환불 영수증 출력 productR productList[] (구조체 배열) typedef struct productR { char name[15]; int price; int quantity; int salePrice; } productR;
POS_PrintReceipt_03	2.5.3.3 Print Receipt_SettlementReceipt	정산 영수증 출력 stockE productList[] (구조체배열) typedef struct StockE { char name[15]; int price; char barcode[5]; int quantity; } stockE;
POS_DB_Update_01	2.6.1.1 DB Update_Sales	환불 정보 업데이트 int s[7] (물품별 환불 갯수들을 담고 있음)
POS_DB_Update_02	2.6.1.2 DB Update_Repayment	판매 정보 업데이트 int s[7] (물품별 판매 갯수들을 담고 있음)

7.5 Feature pass/fail criteria

각 모듈은 SRA 에 정의되어 있는 요구사항(입력/출력 및 동작)을 모두 만족해야 한다. Test 는 normal 값에 대해서만 진행한다. 테스트의 내용은 SRA 의 process description 항목을 참조한다.

8 Unit test case specification

8.1 Test case specification identifier

<Table 4>

Identifier	Input Specification	Output Specification
POS_CashierDisplay_01	판매 캐셔 화면 productNow productList[] (구조체 배열) typedef struct productNow { char name[15]; int price; int quantity; } productNow;	total : 20000 cash : 30000 change : 10000
POS_CashierDisplay_02	환불 캐셔화면 productR list[],int barcode (구조체배열 ,바코드정보) typedef struct productR { char name[15]; int price; int quantity; int salePrice;	date : YYMMDDhhmm refund total : 20000

	} productR;	
POS_Controller_Sales_01	menu==1	salesProcess(); 실행
POS_Controller_Repayment_01	menu==2	repaymentProcess(); 실행
POS_Controller_Stock_01	menu==3	checkStockProcess(); 실행
POS_Controller_Quantity_01	NOTIMPLEMENTED	
POS_Power_01	전원 켜기 power==1	
POS_Customer_01	고객 화면 출력(판매) productNow productList[] (구조체 배열) typedef struct productNow { char name[15]; int price; int quantity; } productNow;	total : 20000 cash : 30000 change : 10000 List : 상품리스트
POS_Customer_02	고객 화면 출력(환불) productR list[],int barcode	date : YYMMDDhhmm refund total : 20000

	(구조체배열 ,바코드정보) typedef struct productR { char name[15]; int price; int quantity; int salePrice; } productR;	
POS_Sales_01	상품 입력 화면 char barcodeData[5] barcodeData == 001/010/011/100/101/110/111 input : barcodeData =001	Valid : printf("enter 1 to proceed Payment. enter 2 to continue scanning ₩n"); Invalid : printf("Invalid Barcode ₩n")
POS_Repayment_01	바코드입력 int barcode	
POS_Repayment_02	바코드유효성 확인 int barcode	Valid : checkBracodeValid(barcode) ==1 Invalid : printf("Wrong Barcode ₩n");
POS_PrintReceipt_01	판매영수증출력 productNow productList[] (구조체 배열) typedef struct productNow {	receipt number : 188373 Date : YYMMDD Sale Product chip 1000 2 2000 icecream 1500 1 1500

	char name[15]; int price; int quantity; } productNow;	fruit 3000 1 3000 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 total : 6500
POS_PrintReceipt_02	환불영수증 출력 productR productList[] (구조체 배열) typedef struct productR { char name[15]; int price; int quantity; int salePrice; } productR;	receipt number : 86896 Date : YYMMDD Sale Product water 500 2 1000 ramen 800 1 800 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 refund total : 1800
POS_PrintReceipt_03	재고확인영수증 출력 stockE productList[] (구조체배열) typedef struct StockE { char name[15]; int price; char barcode[5]; int quantity; } stockE;	Date : YYMMDD chip 1000 98 icecream 1500 99 fruit 3000 99 water 500 98 ramen 800 99 drink 1200 100 coffee 2000 100
POS_DB update_01_Sales	DB 업데이트	DB 재고 업데이트

	int s[7] (물품별 판매 갯수들을 담고 있음) s[7] = {1,1,1,0,0,0,0}	chip 1000 001 99 icecream 1500 010 99 fruit 3000 011 99 water 500 100 100 ramen 800 101 100 drink 1200 110 100 coffee 2000 111 100
POS_DB update_01_Repayment	DB 업데이트 int s[7] (물품별 환불 갯수들을 담고 있음) s[7] = {1,1,1,0,0,0,0}	DB 재고 업데이트 chip 1000 001 99 icecream 1500 010 99 fruit 3000 011 99 water 500 100 100 ramen 800 101 100 drink 1200 110 100 coffee 2000 111 100

8.2 Test items

<Table 3> 참조

8.3 Input specifications

<Table 4> 참조

8.4 Output specifications

<Table 4> 참조

9 Testing tasks

<Table 5 Testing tasks & Schedule >

Task	Predecessor task	Special Skills	Effort	Finish date
Unit Test Plan 작성	SRA 작성 SDS 작성 UTP 작성	POS system 에 대한 이해		2017-11-04
Test Design Specification	Task1	POS system 에 대한 이해		2017-11-05

Test Case Specification	Task2	POS system 에 대한 이해		2017-11-05
Test Execution	Task3	POS system 에 대한 이해 Test Code 작성 Unit Test Tool 활용		2017-11-06
Test result report	Task4	POS system 에 대한 이해		2017-11-06

10 Environmental needs

Point of Sales System 의 Unit Test 를 위한 환경적 요구사항은 다음과 같다.

(1) Platform

- GCC compiler/linker

(2) CTIP(Continuous Testing & Integrated Platform)

- Linux

11 Unit Test deliverables

12 Schedules

<Table 5 Testing tasks & Schedule> 참조