Unit Test Report

for Point Of Sale System

- **Test Cases Specification**
- **Test Summary Report**

Project Team
T6

Latest update on:
2016-11-06

Team Information
201311264 김병식
201610379 김나연
201611248 강병성
201610070 김지우

Table of Contents

1 Introduction

1.1 Objectives

본 문서는 2017년 건국대학교의 소프트웨어공학 개론 강의의 실습과제를 설명한다. 실습 과제는 Point Of Sale (POS) System을 소프트웨어만을 이용한 가상의 시스템을 unit 단위로 구현하여 unit testing 하기 위한 계획 문서이다. Test 수행을 위한 testing Pass/Fail Criteria를 정의하고 이를 수행하기 위한 test design & test cases를 제작한 것들을 실제 테스팅 툴을 사용하여 결과를 기록한다.

1.2 References

[2017SE_B][TP#2]SRA_T6_ver3

[2017SE_B][TP#2]SDS_T6_ver2

[2017SE_B][TP#3]UTP_T6_ver1

2 Unit test case specification

2.1 Test case specification identifier

Identifier	Input	Expected Output
Control Suite		
POS_UTC001	2.1.1의 POS control에 관한 Test Suite이다.	
POS_UTC001_001	SETTLE_SIGN = 0; key = (any key value except for SETTLE);	SETTLE_SIGN → 1 key → SETTLE
POS_UTC001_002	key = TERMINATE;	프로그램 종료
POS_UTC001_003	key = SALE인 상황에 대한 Test Suite	

POS_UTC001_003_C	barcode = 1;
	control = ADD;
	su->items[0].count == 1
		su->items[k].count
		→ su->items[k].count + 1
		(0 ≤ K < 7)

	barcode = 2;
	control = ADD;
	su->items[1].count == 1	

	barcode = 3;
	control = ADD;
	su->items[2].count == 1	

	barcode = 4;
	control = ADD;
	su->items[3].count == 1	

	barcode = 5;
	control = ADD;
	su->items[4].count == 1	

	barcode = 6;
	control = ADD;
	su->items[5].count == 1	

	barcode = 7;
	control = ADD;
	su->items[6].count == 1	

POS_UTC001_003_V	barcode = 1;
	control = ADD;
	innerStorage->list[0].volume == 1
		innerStorage->list[k].volume
		→ innerStorage->list[k].volume + 1
		(0 ≤ K < 7)

	barcode = 2;
	control = ADD;
	innerStorage->list[1].volume == 1	

	barcode = 3;
	control = ADD;
	innerStorage->list[2].volume == 1	

	barcode = 4;
	control = ADD;
	innerStorage->list[3].volume == 1	

	barcode = 5;
	control = ADD;
	innerStorage->list[4].volume == 1	

	barcode = 6;
	control = ADD;
	innerStorage->list[5].volume == 1	

	barcode = 7;
	control = ADD;
	innerStorage->list[6].volume == 1	

POS_UTC001_003_1S	barcode = 1;
	control = ADD;
	innerStorage->total_sales == 1000
		innerStorage->total_sales
		→ innerStorage->total_sales + su->items[0].price

POS_UTC001_003_2S	barcode = 2;	innerStorage->total_sales

	control = ADD; innerStorage->total_sales == 1500	→ innerStorage->total_sales + su->items[1].price

POS_UTC001_003_3S	barcode = 3;
	control = ADD;
	innerStorage->total_sales == 3000
		innerStorage->total_sales
		→ innerStorage->total_sales + su->items[2].price

POS_UTC001_003_4S	barcode = 4;
	control = ADD;
	innerStorage->total_sales == 500
		innerStorage->total_sales
		→ innerStorage->total_sales + su->items[3].price

POS_UTC001_003_5S	barcode = 5;
	control = ADD;
	innerStorage->total_sales == 800
		innerStorage->total_sales
		→ innerStorage->total_sales + su->items[4].price

POS_UTC001_003_6S	barcode = 6;
	control = ADD;
	innerStorage->total_sales == 1200
		innerStorage->total_sales
		→ innerStorage->total_sales + su->items[5].price

POS_UTC001_003_7S	barcode = 7;
	control = ADD;
	innerStorage->total_sales == 2000
		innerStorage->total_sales
		→ innerStorage->total_sales + su->items[6].price

POS_UTC001_003_1A	barcode = 1;
	control = ADD;
	su->amount == 1000
		su->amount → su->amount + su->items[0].price

POS_UTC001_003_2A	barcode = 2;
	control = ADD;
	su->amount == 1500
		su->amount → su->amount + su->items[1].price

POS_UTC001_003_3A	barcode = 3;
	control = ADD;
	su->amount == 3000
		su->amount → su->amount + su->items[2].price

POS_UTC001_003_4A	barcode = 4;
	control = ADD;
	su->amount == 500
		su->amount → su->amount + su->items[3].price

POS_UTC001_003_5A	barcode = 5;
	control = ADD;
	su->amount == 800
		su->amount → su->amount + su->items[4].price

POS_UTC001_003_6A	barcode = 6;
	control = ADD;
	su->amount == 1200
		su->amount → su->amount + su->items[5].price

POS_UTC001_003_7A	barcode = 7; control = ADD;	su->amount → su->amount + su->items[6].price

	su->amount == 2000	

POS_UTC001_003_E	barcode = 1;
	control = ADD;
	innerStorage->list[0].volume = 0;
		key → ERROR

	barcode = 2;
	control = ADD;
	innerStorage->list[1].volume = 0;	

	barcode = 3;
	control = ADD;
	innerStorage->list[2].volume = 0;	

	barcode = 4;
	control = ADD;
	innerStorage->list[3].volume = 0;	

	barcode = 5;
	control = ADD;
	innerStorage->list[4].volume = 0;	

	barcode = 6;
	control = ADD;
	innerStorage->list[5].volume = 0;	

	barcode = 7;
	control = ADD;
	innerStorage->list[6].volume = 0;	

POS_UTC001_003_2E	barcode = (0 ≤ K < 7)이외의 정수 control = ADD;	key → ERROR
POS_UTC001_004	key = REFUND인 상황에 대한 Test Suite	
POS_UTC001_004_V	receipt = 유효한 영수증 번호 innerStorage.list[0].volume == 100	innerStorage->list[k].volume → innerStorage->list[k].volume+1 (0 ≤ K < 7)
	receipt = 유효한 영수증 번호 innerStorage.list[1].volume == 100	
	receipt = 유효한 영수증 번호 innerStorage.list[2].volume == 100	
	receipt = 유효한 영수증 번호 innerStorage.list[3].volume == 100	
	receipt = 유효한 영수증 번호 innerStorage.list[4].volume == 100	
	receipt = 유효한 영수증 번호 innerStorage.list[5].volume == 100	

	receipt = 유효한 영수증 번호 innerStorage.list[6].volume == 100	
POS_UTC001_004_F	receipt = 유효한 영수증 번호 sm->slist[0].sale->flag == '1'	sm->slist[k].sale->flag → '1' (0 ≤ K < 7)
	receipt = 유효한 영수증 번호 sm->slist[1].sale->flag == '1'	
	receipt = 유효한 영수증 번호 sm->slist[2].sale->flag == '1'	
	receipt = 유효한 영수증 번호 sm->slist[3].sale->flag == '1'	
	receipt = 유효한 영수증 번호 sm->slist[4].sale->flag == '1'	
	receipt = 유효한 영수증 번호 sm->slist[5].sale->flag == '1'	
	receipt = 유효한 영수증 번호 sm->slist[6].sale->flag == '1'	
POS_UTC001_004_1T	receipt = 유효한 영수증 번호 innerStorage.total_refund == 1000	innerStorage.total_refund → innerStorage.total_refund + sm->slist[0].sale->amount;
POS_UTC001_004_2T	receipt = 유효한 영수증 번호 innerStorage.total_refund == 1500	innerStorage.total_refund → innerStorage.total_refund + sm->slist[1].sale->amount;
POS_UTC001_004_3T	receipt = 유효한 영수증 번호 innerStorage.total_refund == 3000	innerStorage.total_refund → innerStorage.total_refund + sm->slist[2].sale->amount;
POS_UTC001_004_4T	receipt = 유효한 영수증 번호 innerStorage.total_refund == 500	innerStorage.total_refund → innerStorage.total_refund + sm->slist[3].sale->amount;
POS_UTC001_004_5T	receipt = 유효한 영수증 번호 innerStorage.total_refund == 800	innerStorage.total_refund → innerStorage.total_refund + sm->slist[4].sale->amount;
POS_UTC001_004_6T	receipt = 유효한 영수증 번호 innerStorage.total_refund == 1200	innerStorage.total_refund → innerStorage.total_refund + sm->slist[5].sale->amount;
POS_UTC001_004_7T	receipt = 유효한 영수증 번호 innerStorage.total_refund == 2000	innerStorage.total_refund → innerStorage.total_refund + sm->slist[6].sale->amount;

POS_UTC001_004_C	receipt = 유효한 영수증 번호 sm->rcount == 1	sm->rcount → sm->rcount +1
POS_UTC001_004_E	sm->slist[k].sale->flag = '1' (0 ≤ K < sm->scount) *이미 환불 완료된 영수증 입력상황을 말함	key → ERROR
POS_UTC001_004_CS	receipt = 유효한 영수증 번호 ru->receipt.code.string이 YYYYMMDDHHmm의 포맷 코드인지 확인	ru->receipt.code.string → "YYYYMMDDHHHmm"
POS_UTC001_005	key = SETTLE인 상황에 대한 Test Suite	
POS_UTC001_005_V	innerStorage.list[0].volume == 100	innerStorage->list[k].volume → 100 (0 ≤ K < 7)
	innerStorage.list[1].volume == 100	
	innerStorage.list[2].volume == 100	
	innerStorage.list[3].volume == 100	
	innerStorage.list[4].volume == 100	
	innerStorage.list[5].volume == 100	
	innerStorage.list[6].volume == 100	
POS_UTC001_005_S	innerStorage.total_sales == 0	innerStorage->total_sales → 0
POS_UTC001_005_R	innerStorage.total_refund == 0	innerStorage.total_refund → 0
POS_UTC001_006	key = STOCK인 상황에 대한 Test Suite	
POS_UTC001_006_T	innerStorage->time == posTime	TRUE
POS_UTC001_007	key = PAYMENT인 상황에 대한 Test Suite	
POS_UTC001_007_1C	su->money - su->amount == 0	su->change → su->money – su->amount su->code.string → YYYYMMDDHHmm 형태의 문자열 삽입 sm->slist에 item이 하나 추가 sm->scount → sm->scount + 1
POS_UTC001_007_2C		
POS_UTC001_007_3C		
POS_UTC001_007_4C		
POS_UTC001_007_5C		
POS_UTC001_007_6C		
POS_UTC001_007_7C		
POS_UTC001_007_S	sm->scount == 1	sm->scount → sm->scount + 1
POS_UTC001_007_P	sm->slist is NOT NULL	TRUE

POS_UTC001_007_CS	su->code.string이 YYYYMMDDHHmm의 포맷 코드인지 확인	su->code.string → "YYYYMMDDHHHmm"
Library Suite		
POS_UTC002	2.2의 Touch Screen Interface와 2.3 Customer Screen Interface에 관한 Test Suite이다.	

POS_UTC002_001_1	kb = NULL; su = NULL;
	ru = NULL; innerStorage = NULL;
	cmd = 0;
		캐셔 스크린의 기본 형태를 화면에 출력

POS_UTC002_001_2	kb = NULL; su = NULL;
	ru = NULL; innerStorage = NULL;
	cmd = 1;
		캐셔가 입력할 수 있는 메뉴 창을 출력

POS_UTC002_001_3	kb = NULL; su = NULL;
	ru = NULL; innerStorage = NULL;
	cmd = 2;
		판매/환불 취소 여부를 묻는 메시지를 출력

POS_UTC002_001_4	key = SALE
	kb = NULL; su = NULL;
	ru = NULL; innerStorage = NULL;
	cmd = 3;
		판매 상황에서 캐셔가 입력할 수 있는 메뉴를 출력
		입력 받은 값에 의해
		*kb → ADD/MINUS

POS_UTC002_001_5	key = SALE
	kb = NULL; su에 출력할 상품 판매 리스트 삽입
	ru = NULL; innerStorage = NULL;
	cmd = 4;
		입력 받은 상품을 실시간으로 판매 리스트에 추가, 감소된 화면을 출력

POS_UTC002_001_6	key = PAYMENT
	kb = NULL; su에 출력할 상품 판매 리스트 삽입
	ru = NULL; innerStorage = NULL;
	cmd = 4;
		su의 member를 포맷에 맞춰 출력

POS_UTC002_001_7	key = REFUND;
	kb = NULL; su = NULL;
	ru에 출력할 환불 상품 리스트 삽입 innerStorage = NULL;
	cmd = 3;
		ru의 member를 포맷에 맞춰 출력

POS_UTC002_001_8	key = STOCK;
	kb = NULL; su = NULL;
	ru = NULL; innerStorage에 현재 재고 상태를 삽입
	cmd = 3;
		innerStorage의 member를 포맷에 맞춰 출력

POS_UTC002_001_9	key = CANCEL;
	kb = NULL; su = NULL;
	ru = NULL; innerStorage = NULL;
	cmd = 3;
		판매/환불 동작이 정상적으로 취소됨을 출력

POS_UTC002_001_10	key = ERROR;
	kb = NULL; su = NULL;
	ru = NULL; innerStorage = NULL;
	cmd = 3;
		에러 상황을 출력

POS_UTC002_001_11	key = TERMINATE;
	kb = NULL; su = NULL;
	ru = NULL; innerStorage = NULL;
	cmd = 3;
		포스기가 자원 정리를 정상적으로 마치고 종료됨을 출력

POS_UTC002_002_1	su = NULL; ru = NULL; cmd = 0;	고객 스크린의 기본 형태를 화면에 출력
POS_UTC002_002_2	key = SALE; ru = NULL; cmd = 1; su = su에 출력할 상품 판매 리스트 삽입	입력 받은 상품을 실시간으로 판매 리스트에 추가, 감소된 화면을 출력

POS_UTC002_002_3	key = PAYMENT; ru = NULL;
	cmd = 1;
	su = su에 출력할 상품 판매 리스트 삽입
		su의 member를 포맷에 맞춰 출력

POS_UTC002_002_4	key = REFUND; su = NULL;
	cmd = 1;
	ru = ru에 출력할 환불 상품 리스트 삽입
		ru의 member를 포맷에 맞춰 출력

Print Suite		
POS_UTC003	2.4의 Printer Interface에 관한 Test Suite이다.	

POS_UTC003_1	key = PAYMENT; ru = NULL;
	su = su에 파일로 생성할 상품 판매 리스트 삽입
	innerStorage = NULL; sm = NULL;
		sale_YYYYMMDDHHmm.txt 형태의 파일 생성
		(YYYYMMDDHHmm → su->code.string)

POS_UTC003_2	key = REFUND; su = NULL;	refund_YYYYMMDDHHmm.txt

	ru = ru에 파일로 생성할 환불 상품 리스트 삽입 innerStorage = NULL; sm = NULL;	형태의 파일 생성 (YYYYMMDDHHmm → ru->receipt.code.string)

POS_UTC003_3	key = STOCK; su = NULL; ru = NULL;
	innerStorage = innerStorage에 현재 재고 상태를 삽입
	sm = NULL;
		stock_YYYYMMDD.txt 형태의 파일 생성
		(YYYYMMDD → posTime)

POS_UTC003_4	key = SETTLE; su = NULL; ru = NULL; innerStorage = innerStorage에 현재 재고 상태를 삽입 sm = sm에 하루 동안 남긴 판매 기록에 대한 데이터를 삽입;	settle_YYYYMMDD.txt 형태의 파일 생성 (YYYYMMDD → posTime)
Server Suite		
POS_UTC004	2.5의 Data Access Interface에 관한 Test Suite이다.	
POS_UTC004	innerStorage = innerStorage에 서버와 동기화할 내부 스토리지 상태를 삽입 sm = sm에 서버에 업데이트 할 판매 기록을 삽입	YYYYMMDD_product.txt 형태의 파일 생성 YYYYMMDD_sale_management.txt형태의 파일 생성 (YYYYMMDD → posTime)

2.2 Test items

2.1의 Test case specification을 참조

2.3 Input specifications

2.1의 Test case specification을 참조

2.4 Output specifications

2.1의 Test case specification을 참조

3 Environmental needs

-gcc 5.4.0버전

-cygwin64

-CuTest-1.5

4 Unit test summary report

4.1 Test summary report identifier

Identifier	Result
POS_UTC001_001	OK
POS_UTC001_002	OK
POS_UTC001_003_C	OK
POS_UTC001_003_V	OK
POS_UTC001_003_1S	OK
POS_UTC001_003_2S	OK
POS_UTC001_003_3S	OK
POS_UTC001_003_4S	OK
POS_UTC001_003_5S	OK
POS_UTC001_003_6S	OK
POS_UTC001_003_7S	OK
POS_UTC001_003_1A	OK
POS_UTC001_003_2A	OK
POS_UTC001_003_3A	OK
POS_UTC001_003_4A	OK
POS_UTC001_003_5A	OK
POS_UTC001_003_6A	OK
POS_UTC001_003_7A	OK
POS_UTC001_003_E	OK
POS_UTC001_003_2E	OK
POS_UTC001_004_V	OK
POS_UTC001_004_F	OK
POS_UTC001_004_1T	OK
POS_UTC001_004_2T	OK
POS_UTC001_004_3T	OK
POS_UTC001_004_4T	OK
POS_UTC001_004_5T	OK
POS_UTC001_004_6T	OK
POS_UTC001_004_7T	OK

POS_UTC001_004_C	OK
POS_UTC001_004_E	OK
POS_UTC001_004_CS	OK
POS_UTC001_005_V	OK
POS_UTC001_005_S	OK
POS_UTC001_005_R	OK
POS_UTC001_006_T	OK
POS_UTC001_007_1C	OK
POS_UTC001_007_2C	OK
POS_UTC001_007_3C	OK
POS_UTC001_007_4C	OK
POS_UTC001_007_5C	OK
POS_UTC001_007_6C	OK
POS_UTC001_007_7C	OK
POS_UTC001_007_S	OK
POS_UTC001_007_P	OK
POS_UTC001_007_CS	OK
POS_UTC002_001_1	OK
POS_UTC002_001_2	OK
POS_UTC002_001_3	OK
POS_UTC002_001_4	OK
POS_UTC002_001_5	OK
POS_UTC002_001_6	OK
POS_UTC002_001_7	OK
POS_UTC002_001_8	OK
POS_UTC002_001_9	OK
POS_UTC002_001_10	OK
POS_UTC002_001_11	OK
POS_UTC002_002_1	OK
POS_UTC002_002_2	OK
POS_UTC002_002_3	OK
POS_UTC002_002_4	OK
POS_UTC003_1	OK
POS_UTC003_2	OK
POS_UTC003_3	OK
POS_UTC003_4	OK
POS_UTC004	OK

4.2 Evaluation

Total test case : 66

Passed : 66

Failed : 0