

Static Analysis & 2nd System Test

for Distributed Vending Machine(DVM) Class B - T2

T6

201511273 여찬종

201513205 송승현

201612368 이지우

201811301 한지희

CONTENTS

- 01 Specification Review
- 02 Structural Coverage – Jacoco
- 03 Category-Partition Testing
- 04 Brute Force Testing
- 05 Static Analysis
- 06 Overall

01 Specification Review

Document Version

[OOAD 개발팀 Class B - T2]

Stage	Document Name
	DVM_PFR
1000	OOPT Stage1000 Project Planning Ver.4
SRS	SRS Ver.3
2030	OOPT Stage2030 OOA Ver.4
SDS	SDS Ver.3
2040	OOPT Stage2040 OOD Ver.3

1st Cycle Spec. Review 결과

Spec. Review No.17 Fail

Spec. Review에 대한 총 31개의 Issue 중 30개의 Issue Revision 완료 확인

Jira에 등록된 관련 이슈 완료 처리 * Spec. Review 97%(30 / 31 *100) Pass

No	Stage	Page	Name	Review	Result
17	2030 SRS	9 11	Use Case 14 Type	Typical Courses of Events에 System의 Event(Another DVM)만 존재하므로 Hidden으로 수정 필요	Fail

Spec. Review에 대한 총 31개의 Issue 중 30개의 Issue Revision 완료 확인

Jira에 등록된 관련 이슈 완료 처리, Spec. Review 97%(30 / 31 * 100) Pass

2nd Cycle Spec. Review

1st Cycle Spec. Review No.17

17번 항목에 대한 개발팀과의 논의 결과 Use Case 14의 Type은 Hidden으로 정의 변경된 사항에 따라 6개의 새로운 Issue 등록

No	Stage	Page	Name	Review	Result
1	1000	7	Requirement - R2.3 Give DVM Info	Type을 Hidden으로 수정 필요	Fail
2	1000	10	Use Case - R2.3 Give DVM Info	Type을 Hidden으로 수정 필요	Fail
3	1000	10	Use Case Diagram	Type 이 Hidden 이므로 System 외부의 Another DVM으로의 연결 불필요	Fail
4	2030	9	Use Case 14 Type	Type을 Hidden으로 수정 필요	Fail
5	2030	18	System Sequence Diagram - Use Case 14	System Sequence Diagram 불필요	Fail
6	2040	10	Use Case 14 Type	SRS의 Use Case 14의 Type은 Hidden임으로 통일 필요	Fail

02 Structural Coverage - Jacoco

Jacoco Counter

Unit Test Coverage

Coverage 수식은 (실행된 각 Code단위 수) / (전체 Code단위 수) X 100로 계산함

	1 st Cycle	2 nd Cycle
Instructions Counter	$684/3,586 \times 100 = 19\%$	$740/3,947 \times 100 = 18\%$
Branches Counter	$48/124 \times 100 = 61\%$	$52/138 \times 100 = 62\%$



DVM의 소스 코드가 수정되면서 새로운 Unit Test 작성,

디버깅을 위한 출력 메시지(print문)가 Coverage에 포함되지 않음

해당 내용의 경우 테스트 필요성이 없기 때문에 Issue로 등록하지 않음

Payment, Item Class의 경우 변화 없음

Jacoco

Class	Coverage		
		1 st Cycle	2 nd Cycle
DVM	Instructions	$473/525 \times 100 = 90\%$	$529/560 \times 100 = 94\%$
	Branches	$50/56 \times 100 = 90\%$	$70/78 \times 100 = 89\%$
Payment	Instructions	$90/90 \times 100 = 100\%$	$90/90 \times 100 = 100\%$
	Branches	$12/12 \times 100 = 100\%$	$12/12 \times 100 = 100\%$
Item	Instructions	$58/58 \times 100 = 100\%$	$58/58 \times 100 = 100\%$
	Branches	$4/4 \times 100 = 100\%$	$4/4 \times 100 = 100\%$

03 Category-Partition Testing

Categorize

Phase No.	Phase
1	Activate_Choices
2	Admin_Choices
3	Admin_Login
4	Change_Cash
5	Choose_Item
6	Manage_Amount
7	Not_In_Stock_select
8	Pay By Card
9	Pay by Cash
10	Pay By Code
11	Payment Choices
12	Multiple_action

Phase 12. Multiple_action 추가

여러 대의 DVM이 존재할 경우 재고와 보유 현금이 올바르게 변동되는지 확인
DVM간의 상호작용 테스트

Category-Partition Categorize - 12

추가된 Phase 12 Category

No	Phase	Group	Category	Value	Constraint
12	Multiple_action	Environment	DVM	Current DVM	[property CDVM]
				Target DVM	[property TDVM]
		Action	Stock Change	CDVM Item Sold	[if CDVM] [property CIS]
				TDVM Item Sold	[if TDVM] [property TIS]
			Return Change	Cash Over Input	[if CDVM] [if !CIS !TIS] [property COI]
		Value	Stock	item decrease	[if CIS TIS] [if !COI]
				Cash decrease	[if COI]
			Cash	Cash increase	[if CIS TIS]

Category-Partition Test Result

총 3개의 Test Case가 추가되었으며 모두 Pass 확인

No	Category1 (DVM)	Category2 (Stock Change)	Category3 (Return Change)	Category4 (Stock)	Category5 (Cash)	Test Result
38	Current DVM	CDVM Item Sold	N/A	item decrease	Cash increase	pass
39	Current DVM	N/A	Cash Over Input	N/A	Cash decrease.	pass
40	Target DVM	TDVM Item Sold	N/A	item decrease	Cash increase	pass

* 40개 중 40개 Pass (100%)

04 Brute Force Testing

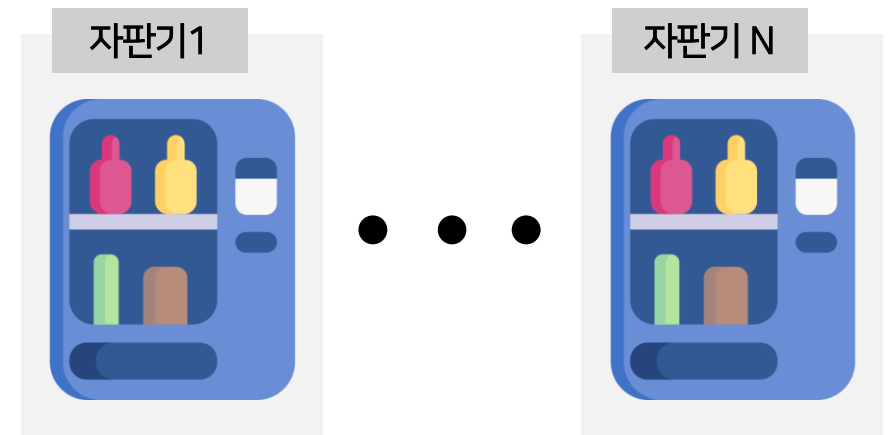
Brute Force Test Case 수정

Description 수정

1st Cycle을 진행하면서 모호한 Description으로 인해 개발팀과의 소통이 원만하게 진행되지 못함
보다 원활한 소통과 Test 진행을 위해 Description을 전반적으로 자세하게 수정

Test Case 삭제

기존 항목에서 비슷한 내용이 중복되어 나타나는 Case를 삭제하고 병합
(기존의 11, 14, 15번 항목)



Brute Force Test Case List - 1

No	Test Case	Description	Test Result
1	DVM 11대 생성 시 생성 불가 메시지 출력	- DVM을 11대까지 생성하는 경우, '자판기는 최대 10개까지만 생성 가능합니다'의 메시지가 출력된다. - '유효하지 않은 DVM'의 메시지 창과 함께 11번째 DVM은 생성되지 않고, 10개의 DVM만 생성된다.	Pass
2	DVM 생성 시 20종류 미만의 음료 설정 시 생성 불가 메시지 출력	- DVM 생성 시 아이템리스트에 20개 미만의 음료 종류를 포함하는 경우, '아이템리스트는 반드시 20개의 아이템을 보유해야 합니다'의 메시지가 출력된다. - '유효하지 않은 DVM'의 메시지 창과 함께 DVM이 생성되지 않는다.	Pass
3	DVM 생성 시 8종류 이상의 음료 재고를 0으로 설정 시 생성 불가 메시지 출력	DVM 생성 시 재고가 0인 음료가 8개 이상인 경우, '재고가 유효한 상품이 7개가 아닌 자판기는 생성할 수 없습니다'의 메시지가 출력된다. - '유효하지 않은 DVM'의 메시지 창과 함께 DVM이 생성되지 않는다.	Pass
4	DVM이 1개일 때, 재고가 없는 음료 선택 시 '재고 없음' 메시지 출력	DVM에서 재고가 없는 음료를 선택하는 경우, '재고가 없는 상품을 선택하여 쿠폰 구입만 가능합니다'의 메시지 창이 출력된다.	Pass
5	DVM이 2개일 때, 모든 DVM에서 재고가 없는 음료 선택 시 '재고 없음' 메시지 출력	- DVM에서 재고가 없는 음료를 선택하는 경우, '재고가 없는 상품을 선택하여 쿠폰 구입만 가능합니다'의 메시지 창이 출력된다. - '다른 DVM확인'을 클릭한 뒤 '근처에 구입가능한 자판기가 없습니다'의 메시지 창 출력을 확인한다.	Pass
6	DVM이 2개일 때, 재고가 0인 DVM에서 재고가 1->0개로 변하는 DVM에 접근 시 '재고 없음' 메시지 출력	- Main DVM에서 재고가 0개인 음료A에 대한 구매를 클릭하고, '재고가 없는 상품을 선택하여 쿠폰 구입만 가능합니다' 메시지 창에서 확인을 누른다. - '다른 DVM확인'을 클릭한 뒤 구입 가능한 자판기에 Another DVM1가 있음을 확인한다. - Another DVM1에서 음료A를 구매한다. - Main DVM에서 '다른 DVM 확인'을 클릭한 뒤 '근처에 구입가능한 자판기가 없습니다'의 메시지 창 출력을 확인한다.	Pass

Brute Force Test Case List - 2

No	Test Case	Description	Test Result
7	DVM이 2개일 때, 재고가 0인 DVM에서 재고가 0->1개로 변하는 DVM에 접근 시 거리 확인 가능	<ul style="list-style-type: none"> - Main DVM에서 재고가 0개인 음료A에 대한 구매를 클릭하고, '재고가 없는 상품을 선택하여 쿠폰 구입만 가능합니다' 메시지 창에서 확인을 누른다. - '다른 DVM확인'을 클릭한 뒤 구입 가능한 자판기에 Another DVM1가 없음을 확인한다. - Another DVM1에서 음료A에 대한 재고를 변경한다. - Main DVM에서 '다른 DVM 확인'을 클릭한 뒤 'Another DVM1'의 거리 정보에 대한 메시지 창 출력을 확인한다. 	Pass
8	DVM이 1개일 때, 카드에 대한 한도 초과 확인	<ul style="list-style-type: none"> - 한도가 4000인 카드로 DVM에서 4000만큼의 음료를 구매한다. - 1000인 음료를 구매했을 때, '카드 한도 초과입니다' 메시지 창을 확인한다. 	Pass
9	DVM이 2개일 때, 카드에 대한 한도 초과 확인	<ul style="list-style-type: none"> - 한도가 4000인 카드로 Main DVM에서 2000, Another DVM1에서 2000만큼의 음료를 구매한다. - Main DVM에서 1000인 음료를 구매했을 때, '카드 한도 초과입니다' 메시지 창을 확인한다. 	Pass
10	DVM이 1개일 때, 음료 구매 시 현금보유량 변화 확인	관리자 메뉴에서 자판기의 현금보유량을 500으로 설정한 뒤 가격이 1000인 음료를 구매했을 때, 현금보유량의 증가를 확인한다.	Pass
11	DVM이 2개일 때, 인증 코드를 통한 구매 가능여부 확인	<ul style="list-style-type: none"> - Main DVM에서 재고가 없는 음료에 대한 인증 코드를 구입한 뒤 인증 코드를 메시지 창에서 확인한다. - 확인한 인증 코드를 Another DVM1의 'Main menu - 쿠폰 사용'을 통해 입력한 뒤, 성공적으로 구매됨을 확인한다. 	Pass
12	DVM에서 반복적으로 음료를 구매하는 경우, 오류가 발생하는지 확인	DVM에서 특정 음료에 대한 구매를 재고 수량만큼 반복하는 과정에서 오류가 발생하는 지 확인한다.	Pass

* 12개 중 12개 Pass (100%): 기존에 Fail 했던 항목에 대해서도 Pass 확인

05 Static Analysis

Maintainability

The image displays three screenshots from an IDE showing maintainability metrics for different classes. Each screenshot includes a sidebar with an overview of metrics and a main window showing the code with a highlighted class.

- Top Left Screenshot (DVM):** The sidebar shows 28 Code Smells, 3h 2min Debt, 2.8% Debt Ratio, and an Overall Rating of A. The code window shows the `DVM` class with 28 Code Smells highlighted.
- Top Right Screenshot (Payment):** The sidebar shows 4 Code Smells, 47min Debt, 2.4% Debt Ratio, and an Overall Rating of A. The code window shows the `Payment` class with 4 Code Smells highlighted.
- Bottom Left Screenshot (Item):** The sidebar shows 1 Code Smell, 10min Debt, 0.9% Debt Ratio, and an Overall Rating of A. The code window shows the `Item` class with 1 Code Smell highlighted.

Item (Top Middle)

Item (Bottom Left)

Code Smell 수정 시 총 3H 59M 소요 예상

Cyclomatic Complexity

순환 복잡도

소스 코드의 복잡도를 나타내는 지표

DVM

```
public DVM(String region, int x, int y, int totalCash, String adminID, String adminPW, ArrayList<Item> itemList, int dist, MyPayment myPayment, boolean duplicated) {
    if (DVMList.size() == 10) {
        System.out.println("자판기는 최대 10개까지만 생성 가능합니다.");
        System.out.println("[ " + region + " ] 자판기는 생성되지 않았습니다.");
    } else if (itemList.size() != 20) {
        System.out.println("아이템리스트는 반드시 20개의 아이템을 보유해야 합니다.");
        System.out.println("[ " + region + " ] 자판기는 생성되지 않았습니다.");
    } else {
        int availableItem = 0;
        for (int i=0; i<20; i++) {
            if (itemList.get(i).getItemAmount() > 0) {
                availableItem += 1;
            }
        }
        if (availableItem != 7) {
            System.out.println("재고가 유효한 상품이 개가 아닌 자판기는 생성할 수 없습니다.");
            System.out.println("[ " + region + " ] 자판기는 생성되지 않았습니다.");
        } else {
            this.region = region;
            this.location = new Pair(x,y);
            this.totalCash = totalCash;
            this.adminID = adminID;
            this.adminPW = adminPW;
            this.itemList = itemList;
            this.dist = dist;
            this.myPayment = myPayment;
            boolean duplicated = false;
            for(int i=0;i<DVMList.size();i++){
                if(DVMList.get(i).location.x == x && DVMList.get(i).location.y ==y){
                    duplicated=true;
                }
            }
        }
    }
}
```

Cyclomatic Complexity 162	
AdminMenuGUI.java	5
CardGUI.java	8
CashGUI.java	9
DVM.java	64
Item.java	9
ItemListGUI.java	6
LoginGUI.java	5
Main.java	1
MainGUI.java	
ManageAmountGUI.java	
ManageCashGUI.java	
NotEnoughAmountGUI.java	
Payment.java	13
SelectPaymentGUI.java	
UseCouponGUI.java	

1-10: 단순 프로그램, 위험성 낮음

11-20: 다소 복잡한 프로그램, 위험성 중간

21-50: 복잡한 프로그램, 위험성 중간

51이상: 테스트 불가능한 프로그램, 위험성 매우 높음

DVM.java: 64 / Item.java: 9 / Payment.java: 13

DVM Class의 경우 매우 높은 복잡도, 위험성이 매우 높음



Sonarqube Issue

No	File	Line	Rule	Severity	Description
1	DVM.java	133	Bug	Minor	Math.pow 함수의 입력은 double형이어야 합니다.
2		26	Code Smell	Major	생성자의 파라미터가 너무 많습니다.
3		247	Bug	Major	문자열 비교에 ==을 사용하지 마세요.
4		29	Code Smell	Critical	String이 반복적으로 나타나고 있습니다.
5		26	Code Smell	Critical	조건문과 반복문으로 인해 코드가 복잡합니다.
6		228, 251	Code Smell	Minor	Boolean 대신 boolean을 사용하세요. Boolean 사용 시 NPE 대비할 수 있도록 코드 작성해야 합니다.
7	Payment.java	13	Bug	Minor	Double Brace Initialization은 권장되지 않습니다.






Sonarqube 분석 결과 총 4개의 Bug, 244개의 Code Smell 탐지

GUI 부분을 제외하고, False Alarm을 제외하여 **총 7개의 Issue를 선정하여 등록**

Issue Upload to Jira

프로젝트 /  konkuk_VV_t_6_2021 / KVT62-126 /  KVT62-144






















Static Analysis

 첨부  하위 작업 생성  이슈 연결  

설명
GUI를 제외한 DVM, Payment, Item.java에서의 Static Analysis Issue에 대해서만 Issue를 등록하였습니다.
해당 Issue 클릭 시 Sonarqube로 연결되고, 로그인 후 위치 확인 가능합니다!

링크된 이슈 +

blocks

<input checked="" type="checkbox"/>	KVT62-12 1. Math operands should be cast before assignment			BACKLOG 
<input checked="" type="checkbox"/>	KVT62-131 5. Cognitive Complexity of methods should not be too high			BACKLOG 
<input checked="" type="checkbox"/>	KVT62-25 2. Methods should not have too many parameters			BACKLOG 
<input checked="" type="checkbox"/>	KVT62-127 3. Strings and Boxed types should be compared using "equals()"			BACKLOG 
<input checked="" type="checkbox"/>	KVT62-129 4. String literals should not be duplicated			BACKLOG 
<input checked="" type="checkbox"/>	KVT62-137 6. Boxed "Boolean" should be avoided in boolean expressions			BACKLOG 
<input checked="" type="checkbox"/>	KVT62-13 7. Double Brace Initialization should not be used			BACKLOG 

06 Overall

Overall

1st Cycle 종료 후 전반적인 Quality

	Specification Review	Category Partition Test	Brute Force Test	Static Analysis Issue	Total
Passed / All Test Case	31 / 31	40 / 40	12 / 12	0 / 7	83 / 90

프로젝트 / konkuk_VV_t_6_2021 / KVT62 보드

칸반 보드

내 이슈만 최근 업데이트됨

디버깅이슈 8 스펙리뷰 0 시스템테스팅 0 진행 중 1

급행 4개 이슈

Static Analysis
2nd Cycle
KVT62-144

2nd Cycle
2nd Cycle
KVT62-126

나머지 기타 126개 이슈

1. Math operands should be cast before assignment
KVT62-12

7. Double Brace Initialization should not be used
KVT62-13

2nd Cycle을 진행하며 총 7개의 새로운 이슈가 Jira에 등록됨

1st Cycle이 완료된 후 처리된 이슈들은 Jira에서 완료 처리

Overall

Feedback

- DVM에 대한 기능이 Test가 가능한 범위 내에서 잘 구현되어 있음
 - > 디버깅을 위한 로그 출력문을 제외하면 Unit Test가 모든 코드를 Cover하고 있음
- DVM간의 재고 처리, 쿠폰 구입 등의 상호작용이 문제 없이 잘 구현되어 있음
 - > 상호작용을 테스트 하기 위한 Category-Partition Test 결과 모든 항목에서 Pass