Category Partition Testing Tool

2014114134 오세욱

2014114136 임현유

2012111375 임동현

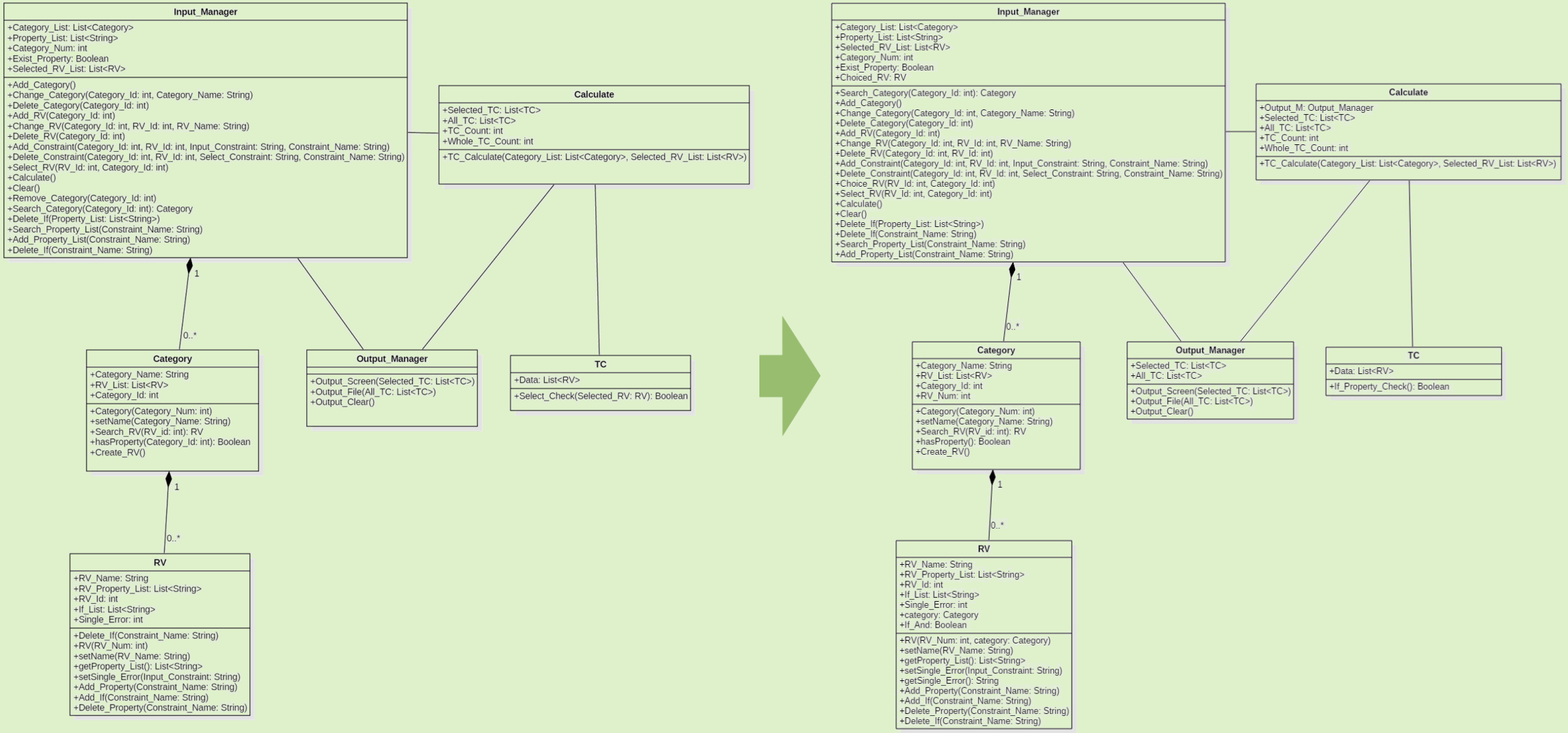
2012111387 하헌규

Index

Revise Plan

1.1 Class Diagram

Revise Plan



1.1 Class Diagram

Revise Plan

1 Input_Manager
- Attribute
  Choiced_RV
- Operation
  Choice_RV 추가
  Remove_Category 삭제

2 Category
- Attribute
  RV_Num 추가

3 RV
- Attribute
  category 추가
  If_And 추가
- Operation
  getSingle_Error 추가

4 Calculate
- Attribute
  Output_M 추가

5 Output_Manager
-Attribute
  Selected_TC 추가
  All_TC 추가

Implement Class & Methods Definitions

Implement Class & Methods Definitions(1/38)

Type	Class
Name	Input_Manager
Purpose	Actor의 Input을 수행하는 클래스
Overview	Category, RV 객체 관리, Test Case 계산, 초기화 등을 제어한다.
Cross Reference	Functions: ALL Use Cases:ALL
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(2/38)

Type	Class
Name	Output_Manager
Purpose	Input에 대한 결과 값을 출력하는 클래스
Overview	Test Case 결과를 File로 출력, Selected_TC를 화면에 출력한다.
Cross Reference	Functions : R2.1, R3.1 Use Cases : TestCase 계산, 초기화
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(3/38)

Type	Class
Name	Calculate
Purpose	TestCase 계산 클래스
Overview	현재 Category, RV의 상태에따라 TestCase를 계산한다.
Cross Reference	Functions : R2.1 Use Cases : TestCase 계산
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(4/38)

Type	Class
Name	Category
Purpose	Category 클래스
Overview	Category 데이터 관리와 관련 기능을 한다.
Cross Reference	Functions : R1.2, R1.4 Use Cases : Category 수정, 대푯값 추가
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(5/38)

Type	Class
Name	RV
Purpose	RV 클래스
Overview	RV 데이터 관리와 관련 기능을 한다.
Cross Reference	Functions : R1.5, R1.7, R1.8 Use Cases : 대푯값 수정, Constraint추가, Costraint 삭제
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(6/38)

Type	Class
Name	TC
Purpose	TC 클래스
Overview	TC데이터 관리와 관련 기능, TestCase 계산 관련 기능을 한다.
Cross Reference	Functions : R2.1 Use Cases : Test Case 계산
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(7/38)

Type	Method
Name	Search_Category
Purpose	전체 Category중 특정 Category를 찾는다.
Cross Reference	Functions : R1.2, R1.4, R1.5, R1.6, R1.7, R1.8 Use Cases : Category 수정, 대푯값 추가, 대푯값 수정, 대푯값 삭제, Constraint 추가, Constraint 삭제
Input(Method)	int Category_Id
Output(Method)	Category Category
Abstract operation(Method)	Category_Id를 받아서, 일치하는 Category를 반환한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(8/38)

Type	Method
Name	Add_Category
Purpose	Category객체를 추가한다.
Cross Reference	Functions : R1.1 Use Cases : Category 추가
Input(Method)	Void
Output(Method)	Void
Abstract operation(Method)	Category_List에 Category 객체를 추가한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(9/38)

Type	Method
Name	Change_Category
Purpose	Category 객체의 이름을 바꾼다.
Cross Reference	Functions : R1.2 Use Cases : Category 수정
Input(Method)	int Category_Id, String Category_Name
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 객체의 이름을 Category_Name으로 바꾼다.
Exceptional Courses of Events	Category_Name이 “”라면 이름을 변경하지 않는다.

Implement Class & Methods Definitions(10/38)

Type	Method
Name	Delete_Category
Purpose	Category 객체를 삭제한다.
Cross Reference	Functions : R1.3 Use Cases : Category 삭제
Input(Method)	int Category_Id
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 Category 객체를 삭제한다.
Exceptional Courses of Events	Category에 Property가 존재하면 삭제하지 않는다.

Implement Class & Methods Definitions(11/38)

Type	Method
Name	Add_RV
Purpose	RV 객체를 추가한다.
Cross Reference	Functions : R1.4 Use Cases : 대푯값 추가
Input(Method)	int Category_Id
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 Category 객체에 RV 객체를 추가한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(12/38)

Type	Method
Name	Change_RV
Purpose	RV 객체의 이름을 변경한다.
Cross Reference	Functions : R1.5 Use Cases : 대푯값 수정
Input(Method)	int Category_Id, Int RV_Id, String RV_Name
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 Category 객체에서 RV 객체를 찾는다. 3. RV 객체의 이름을 변경한다.
Exceptional Courses of Events	RV_Name이 “”라면 이름을 변경하지 않는다.

Implement Class & Methods Definitions(13/38)

Type	Method
Name	Delete_RV
Purpose	RV 객체를 삭제한다.
Cross Reference	Functions : R1.6 Use Cases : 대푯값 삭제
Input(Method)	int Category_Id, int RV_Id
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 Category 객체에서 RV 객체를 찾는다. 3. 찾는 RV객체를 삭제한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(14/38)

Type	Method
Name	Add_Constraint
Purpose	Constraint를 추가한다.
Cross Reference	Functions : R1.7 Use Cases : Constraint 추가
Input(Method)	int Category_Id, int RV_Id, String Input_Constraint, String Constraint_Name
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 Category 객체에서 RV 객체를 찾는다. 3. 찾는 RV객체에 Input_Constriant와 Constraint_Name에 따라 Constraint를 추가한다.
Exceptional Courses of Events	Property중 Constraint_Name이 존재하지 않는 If Property는 추가하지 않는다.

Implement Class & Methods Definitions(15/38)

Type	Method
Name	Delete_Constraint
Purpose	Constraint를 삭제한다.
Cross Reference	Functions : R1.8 Use Cases : Constraint 삭제
Input(Method)	int Category_Id, int RV_Id, String Select_Constraint, String Constraint_Name
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 Category 객체에서 RV 객체를 찾는다. 3. 찾는 RV객체에 Select_Constriant와 Constraint_Name에 따라 Constraint를 삭제한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(16/38)

Type	Method
Name	Choice_RV
Purpose	선택한 Constraint와 관련된 RV를 기억한다.
Cross Reference	Functions : R1.7, R1.8 Use Cases : Constraint 추가, Constraint 삭제
Input(Method)	int RV_Id, int Category_Id
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 Category 객체에서 RV 객체를 찾는다. 3. 찾는 RV객체를 기억한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(17/38)

Type	Method
Name	Select_RV
Purpose	선택한 RV의 선택 여부를 변경한다.
Cross Reference	Functions : R1.9 Use Cases : 대폿값 선택
Input(Method)	int RV_Id, int Category_Id
Output(Method)	Void
Abstract operation(Method)	1. Category 객체중 Id 가 Category_Id와 일치하는 객체를 찾는다. 2. 찾은 Category 객체에서 RV 객체를 찾는다. 3. 찾는 RV객체의 선택 여부를 변경한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(18/38)

Type	Method
Name	Calculate
Purpose	Test Case를 계산한다.
Cross Reference	Functions : R2.1 Use Cases : Test Case 계산
Input(Method)	Void
Output(Method)	Void
Abstract operation(Method)	Test Case를 계산한다.
Exceptional Courses of Events	RV가 존재하지 않는 Category가 있으면 계산하지 않는다.

Implement Class & Methods Definitions(19/38)

Type	Method
Name	Clear
Purpose	초기화한다.
Cross Reference	Functions : R3.1 Use Cases : 초기화
Input(Method)	Void
Output(Method)	Void
Abstract operation(Method)	1. Output_Manager를 Clear한다. 2. Input_Manager를 Clear한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(20/38)

Type	Method
Name	Delete_If
Purpose	If Constraint를 삭제한다.
Cross Reference	Functions : R1.6, R1.8 Use Cases : 대푯값 삭제, Constraint 삭제
Input(Method)	List<String> Property_List
Output(Method)	Void
Abstract operation(Method)	Property_list에서 Constraint_Name을 찾는다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(21/38)

Type	Method
Name	Search_Property_List
Purpose	Property_List에 Constraint_Name이 존재하는지 확인한다.
Cross Reference	Functions : R1.7 Use Cases : Constraint 추가
Input(Method)	String Constraint_Name
Output(Method)	Boolean
Abstract operation(Method)	Property_list에서 Constraint_Name을 찾는다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(22/38)

Type	Method
Name	Add_Property_List
Purpose	Property_List에 Constrait_Name을 추가한다.
Cross Reference	Functions : R1.7 Use Cases : Constraint 추가
Input(Method)	String Constraint_Name
Output(Method)	void
Abstract operation(Method)	Property_List에 Constraint_Name를 추가한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(23/38)

Type	Method
Name	Output_Screen
Purpose	화면에 선택된 TestCase를 출력한다.
Cross Reference	Functions : R2.1 Use Cases : Test Case 계산
Input(Method)	List<TC> Selected_TC
Output(Method)	void
Abstract operation(Method)	Selected_TC를 받아 화면에 출력한다.
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(24/38)

Type	Method
Name	Output_File
Purpose	결과를 File로 출력
Cross Reference	Functions : R2.1 Use Cases : Test Case 계산
Input(Method)	ArrayList All_TC
Output(Method)	void
Abstract operation(Method)	결과를 out.txt text file로 출력
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(25/38)

Type	Method
Name	Output_Clear
Purpose	출력 부분 초기화
Cross Reference	Functions : R3.1 Use Cases : 초기화
Input(Method)	void
Output(Method)	void
Abstract operation(Method)	ArrayList Selected_TC 초기화
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(26/38)

Type	Method
Name	setName
Purpose	Category의 이름 변경
Cross Reference	Functions : R1.2 Use Cases : Category 수정
Input(Method)	String Category_Name
Output(Method)	void
Abstract operation(Method)	Category의 이름을 Category_Name으로 변경
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(27/38)

Type	Method
Name	Search_RV
Purpose	RV를 Search
Cross Reference	Functions : R1.4, R1.5, R1.6 Use Cases : 대푯값 추가, 대푯값 수정, 대푯값 삭제
Input(Method)	Int RV_id
Output(Method)	RV RV
Abstract operation(Method)	RV_List 를 순회하면서 RV_id와 일치하는 RV를 반환
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(28/38)

Type	Method
Name	hasProperty
Purpose	Category 내에 Property 값이 존재하는지 확인하기 위한 메소드
Cross Reference	Functions : R1.3 Use Cases : Category 삭제
Input(Method)	Int Category_Id
Output(Method)	Boolean
Abstract operation(Method)	RV가 갖는 Property List size가 0이 아니면 True 반환
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(29/38)

Type	Method
Name	Create_RV
Purpose	RV를 추가하기 위한 메소드
Cross Reference	Functions : R1.4 Use Cases : 대푯값 추가
Input(Method)	N/A
Output(Method)	Void
Abstract operation(Method)	RV_List에 RV객체 생성 후 추가
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(30/38)

Type	Method
Name	setName
Purpose	RV_Name 변경하기 위한 메소드
Cross Reference	Functions : R1.5 Use Cases : 대푯값 수정
Input(Method)	String RV_Name
Output(Method)	Void
Abstract operation(Method)	RV_Name을 변경
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(31/38)

Type	Method
Name	getProperty_List
Purpose	현재 RV의 Property list를 반환하기 위한 메소드
Cross Reference	Functions : Use Cases :
Input(Method)	N/A
Output(Method)	List<String>
Abstract operation(Method)	현재 RV의 Property list를 반환
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(32/38)

Type	Method
Name	setSingle_Error
Purpose	Input_Constraint에 따라 Single, Error를 정하는 메소드
Cross Reference	Functions : Use Cases :
Input(Method)	String Input_Constraint
Output(Method)	Void
Abstract operation(Method)	Input_Constraint가 0이면 Single_Error = 0, “Single”이면 Single_Error = 1, “Error”면 Single_Error = 2
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(33/38)

Type	Method
Name	Add_Property
Purpose	현재 RV의 property list에 Property 추가
Cross Reference	Functions : R1.7 Use Cases : Constraint 추가
Input(Method)	String Constraint_Name
Output(Method)	Void
Abstract operation(Method)	RV_Property_List가 입력받은 Constraint_Name을 포함하지 않으면 리스트에 추가
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(34/38)

Type	Method
Name	Add_If
Purpose	현재 RV의 If list에 If추가
Cross Reference	Functions : R1.7 Use Cases : Constraint 추가
Input(Method)	String Constraint_Name
Output(Method)	Void
Abstract operation(Method)	If_List가 입력받은 Constraint_Name을 포함하지 않으면 리스트에 추가
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(35/38)

Type	Method
Name	Delete_Property
Purpose	현재 RV의 property를 삭제
Cross Reference	Functions : R1.8 Use Cases : Constraint 삭제
Input(Method)	String Constraint_Name
Output(Method)	Void
Abstract operation(Method)	RV_Property_List size만큼 순회하면서 Constraint_Name과 같은 Property를 삭제
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(36/38)

Type	Method
Name	Delete_If
Purpose	현재 RV의 If 를 삭제
Cross Reference	Functions : R1.8 Use Cases : Constraint 삭제
Input(Method)	String Constraint_Name
Output(Method)	Void
Abstract operation(Method)	If_List size 만큼 순회하면서 Constraint_Name과 같은 If를 삭제
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(37/38)

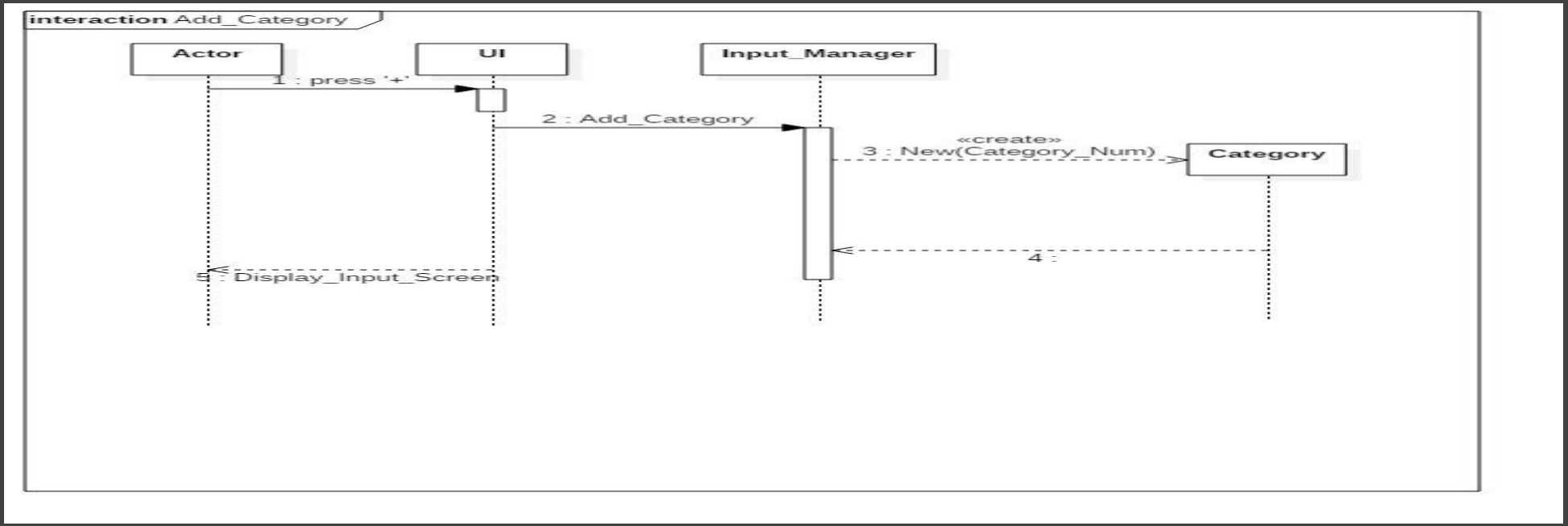
Type	Method
Name	If_Property_Check
Purpose	Property – If 관계를 확인한다.
Cross Reference	Functions : R1.7 R1.9 Use Cases : Constraint 추가, Test Case 계산
Input(Method)	N/A
Output(Method)	Boolean
Abstract operation(Method)	If가 갖는 이름에 해당하는 Property 이름이 존재하면 True 반환
Exceptional Courses of Events	N/A

Implement Class & Methods Definitions(38/38)

Type	Method
Name	TC_Calculate
Purpose	테스트 케이스를 계산한다.
Cross Reference	Functions : R2.1 Use Cases : Test Case 계산
Input(Method)	ArrayList Category_List, ArrayList Selected_RV_List
Output(Method)	void
Abstract operation(Method)	Category_List와 Selected_RV_List로 Test Case를 계산한다.
Exceptional Courses of Events	N/A

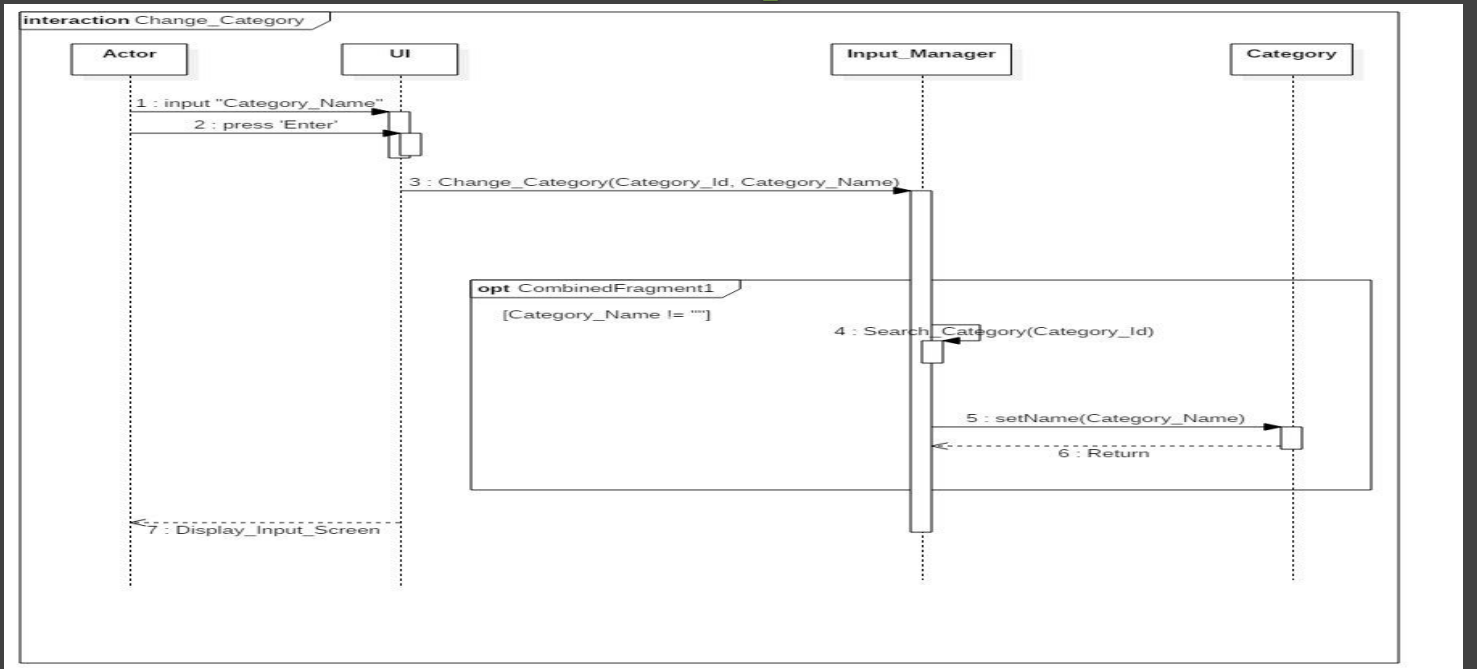
Implement Windows

Implement Windows(1/13)



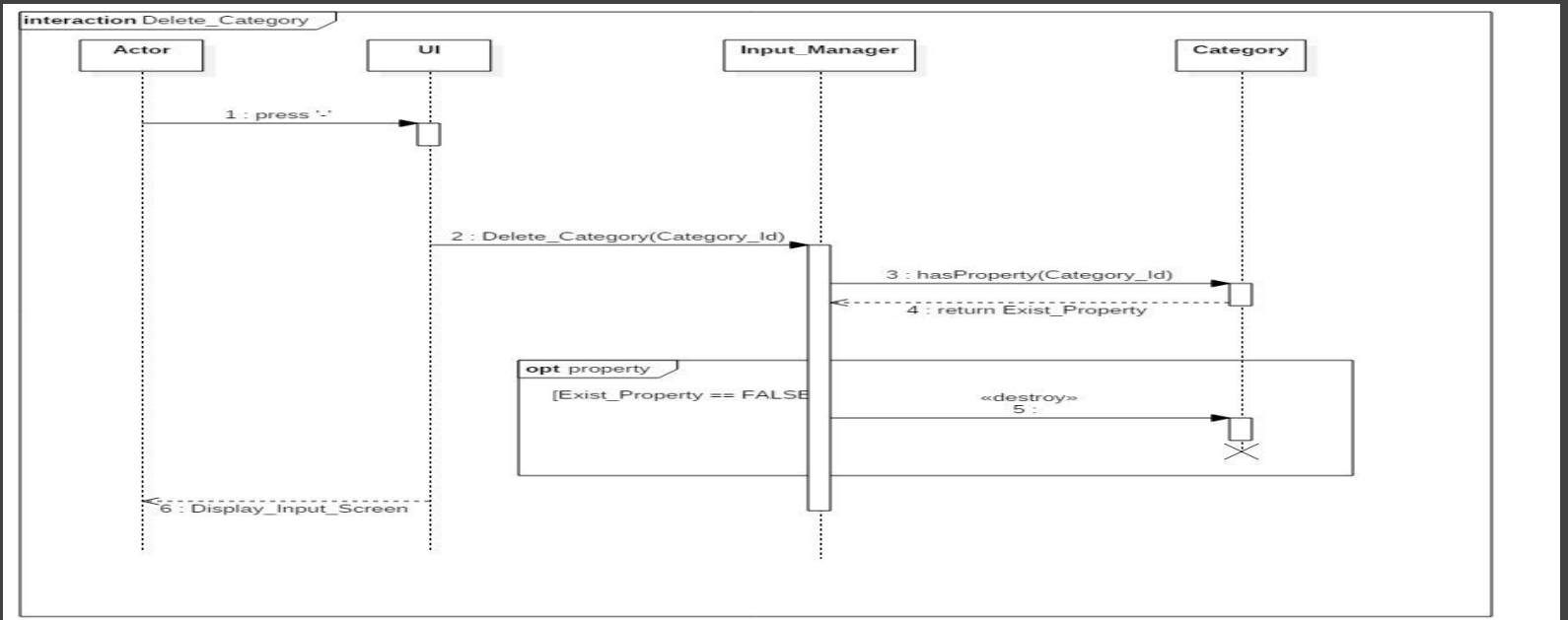
Name	Add_CategoryButton
Responsibilities	Category 추가 버튼을 클릭한다.
Type	GUI
Cross Reference	R1.1
Notes	Category를 추가해주는 Add_Category()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Add_Category()가 실행된다.

Implement Windows(2/13)



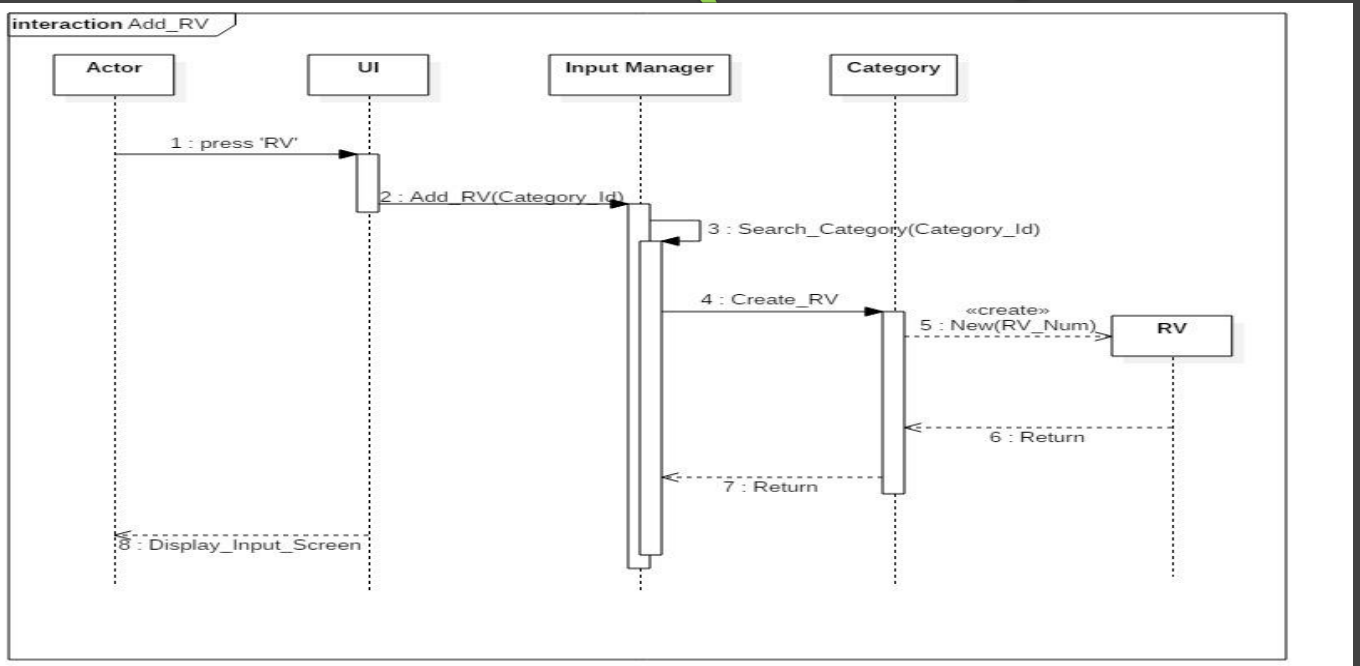
Name	CategorytextField
Responsibilities	Category 이름과 Enter 입력한다.
Type	GUI
Cross Reference	R1.2
Notes	Category이름을 변경해주는 Change_Category()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Change_Category()가 실행된다.

Implement Windows(3/13)



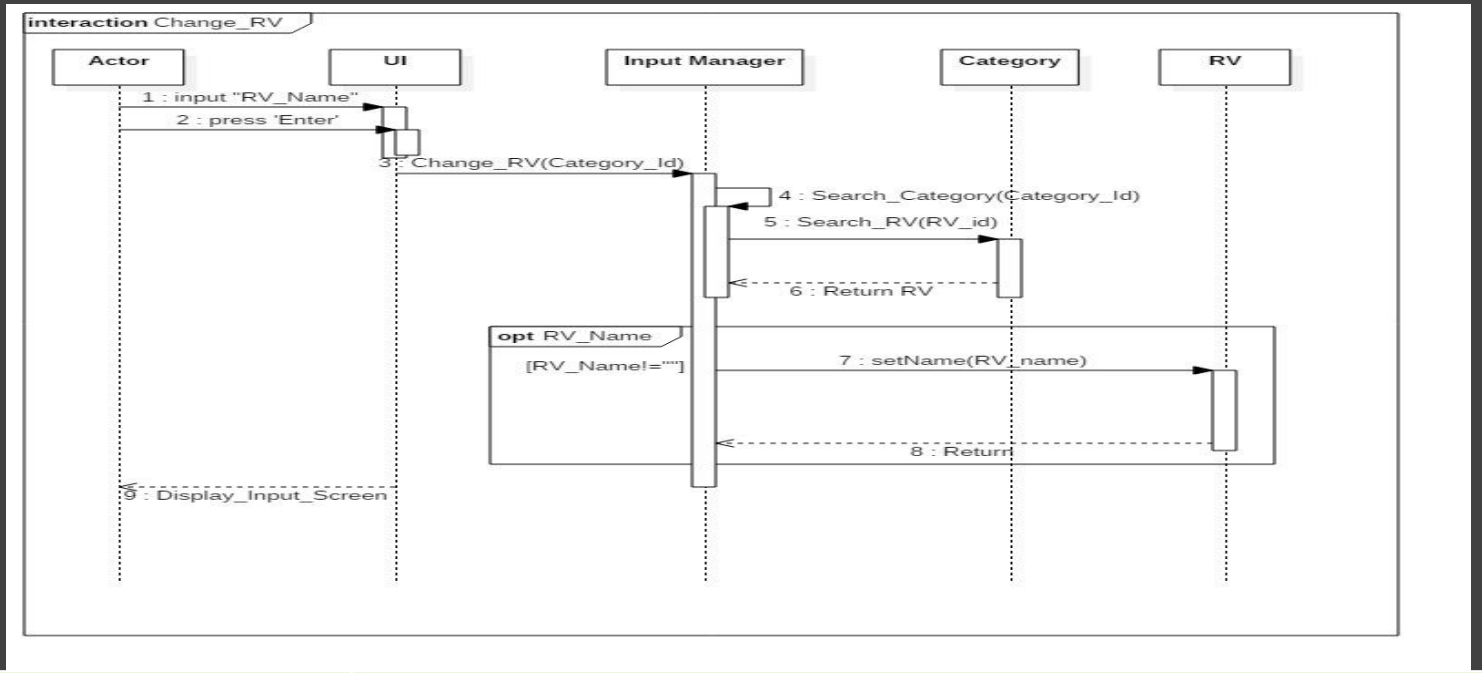
Name	Delete_CategoryButton
Responsibilities	Category 삭제 버튼을 클릭한다.
Type	GUI
Cross Reference	R1.3
Notes	Category를 삭제해주는 Delete_Category()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Delete_Category()가 실행된다.

Implement Windows(4/13)



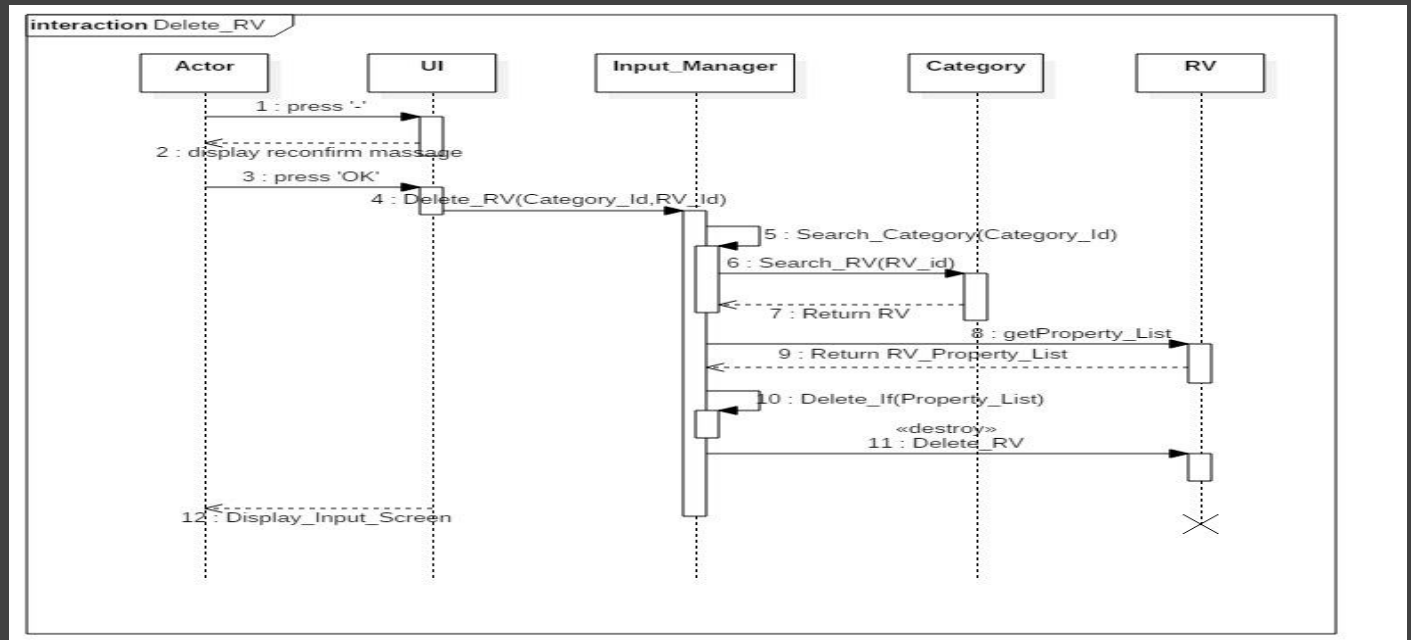
Name	Add_RVButton
Responsibilities	RV 추가 버튼을 클릭한다.
Type	GUI
Cross Reference	R1.4
Notes	RV를 추가해주는 Add_RV()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Add_RV()가 실행된다.

Implement Windows(5/13)



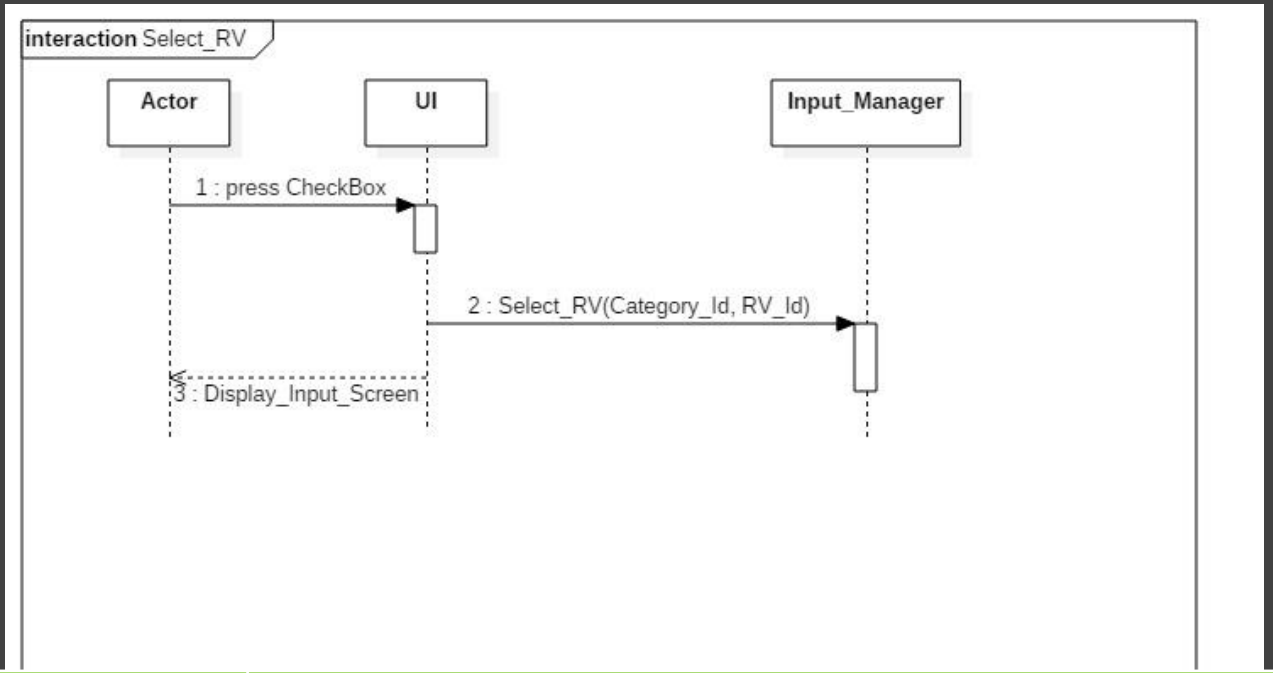
Name	RVtextField
Responsibilities	RV의 이름과 Enter를 입력한다.
Type	GUI
Cross Reference	R1.5
Notes	RV의 이름을 변경해주는 Change_RV()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Change_RV()가 실행된다.

Implement Windows(6/13)



Name	Delete_RVButton
Responsibilities	RV 삭제 버튼을 클릭한다.
Type	GUI
Cross Reference	R1.6
Notes	RV를 삭제해주는 Delete_RV()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Delete_RV()가 실행된다.

Implement Windows(7/13)

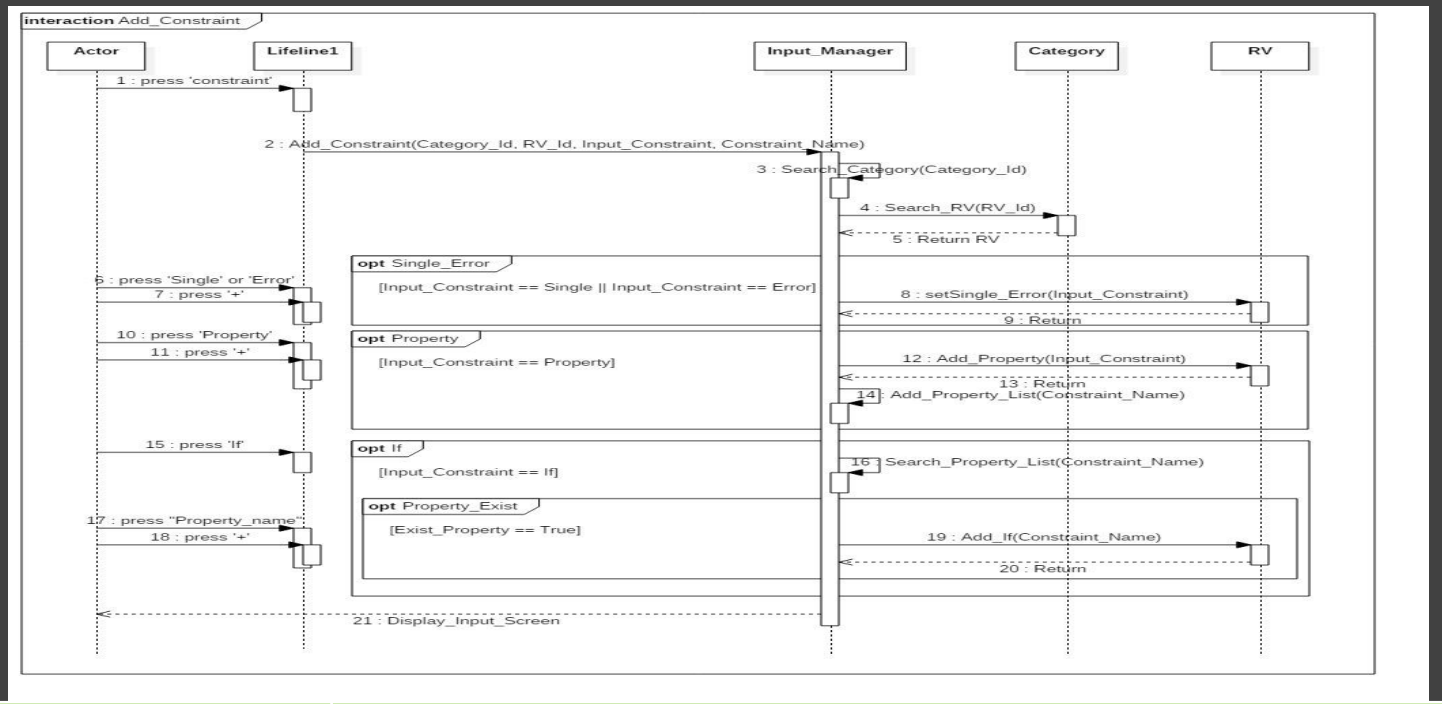


Name	checkbox
Responsibilities	checkbox를 클릭한다.
Type	GUI
Cross Reference	R1.9
Notes	RV의 선택 여부를 변경해주는 Select_RV()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Select_RV()가 실행된다.

Implement Windows(8/13)

Name	chooseRVbutton
Responsibilities	chooseRV버튼을 클릭한다.
Type	GUI
Cross Reference	R1.7
Notes	Constraint를 추가할 RV를 선택한다.
Pre-Conditions	N/A
Post-Conditions	선택한 RV가 저장된다.

Implement Windows(9/13)

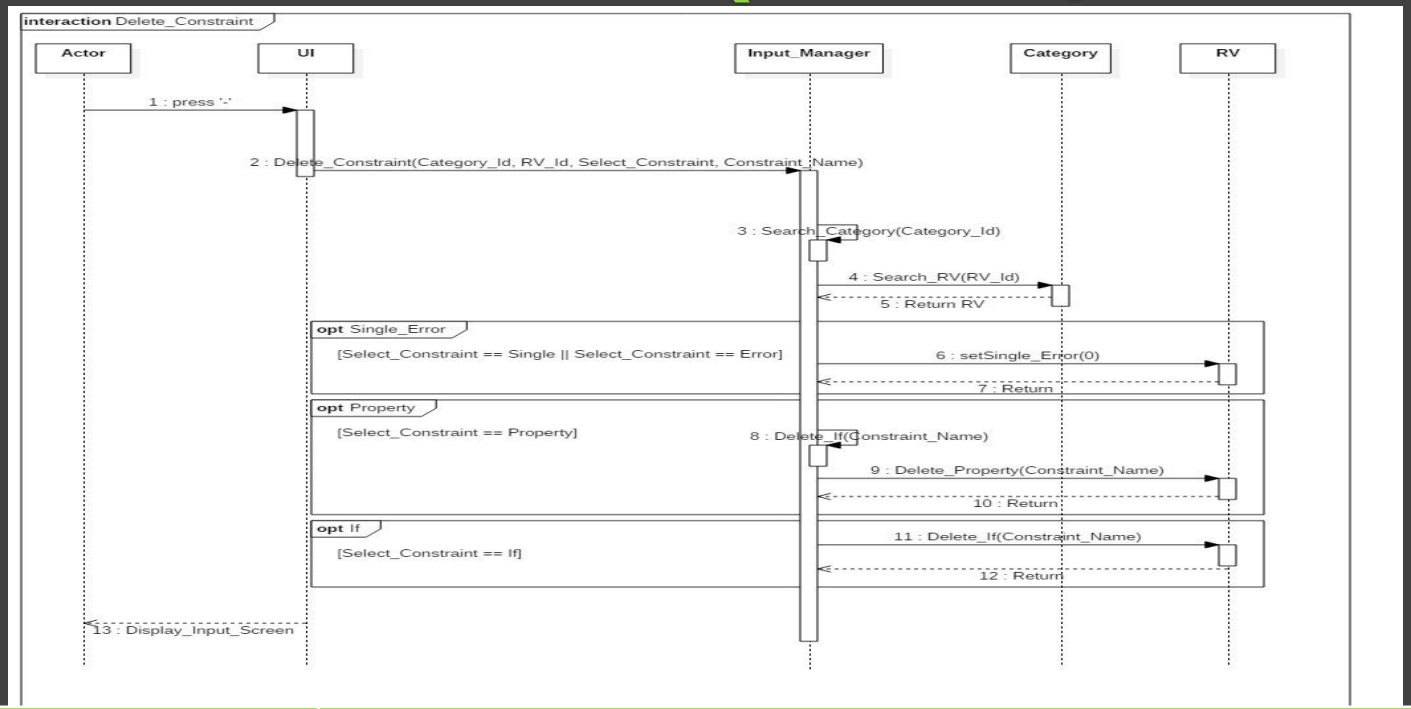


Name	selectConstraintButton
Responsibilities	selectConstraintButton을 클릭하고 Constraint를 선택한다.
Type	GUI
Cross Reference	R1.7
Notes	선택한 Constraint를 반환한다.
Pre-Conditions	chooseRVbutton을 클릭해야 한다.
Post-Conditions	선택한 Constraint를 저장한다.

Implement Windows(10/13)

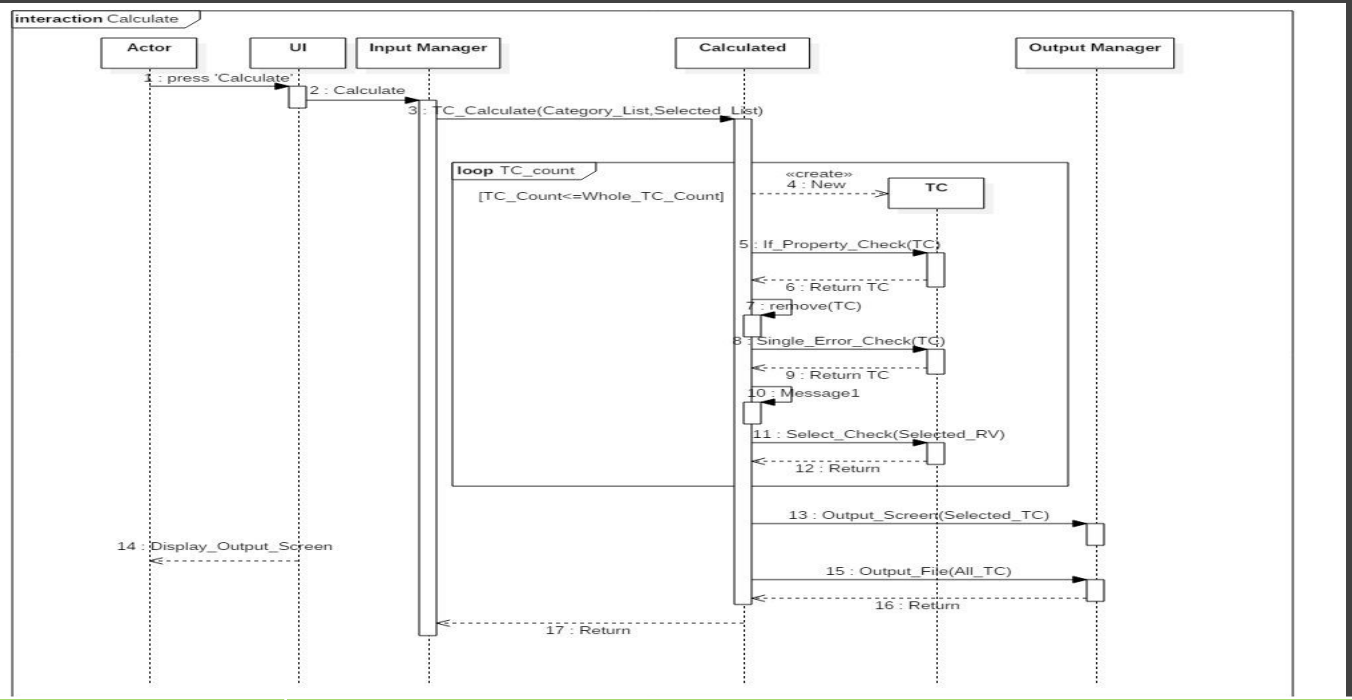
Name	addConstraintButton
Responsibilities	Constraint 추가 버튼을 클릭한다.
Type	GUI
Cross Reference	R1.7
Notes	Constraint를 추가해주는 Add_Constraint()메서드를 호출한다.
Pre-Conditions	Constraint를 선택해야한다.
Post-Conditions	Add_Constraint()가 실행된다.

Implement Windows(11/13)



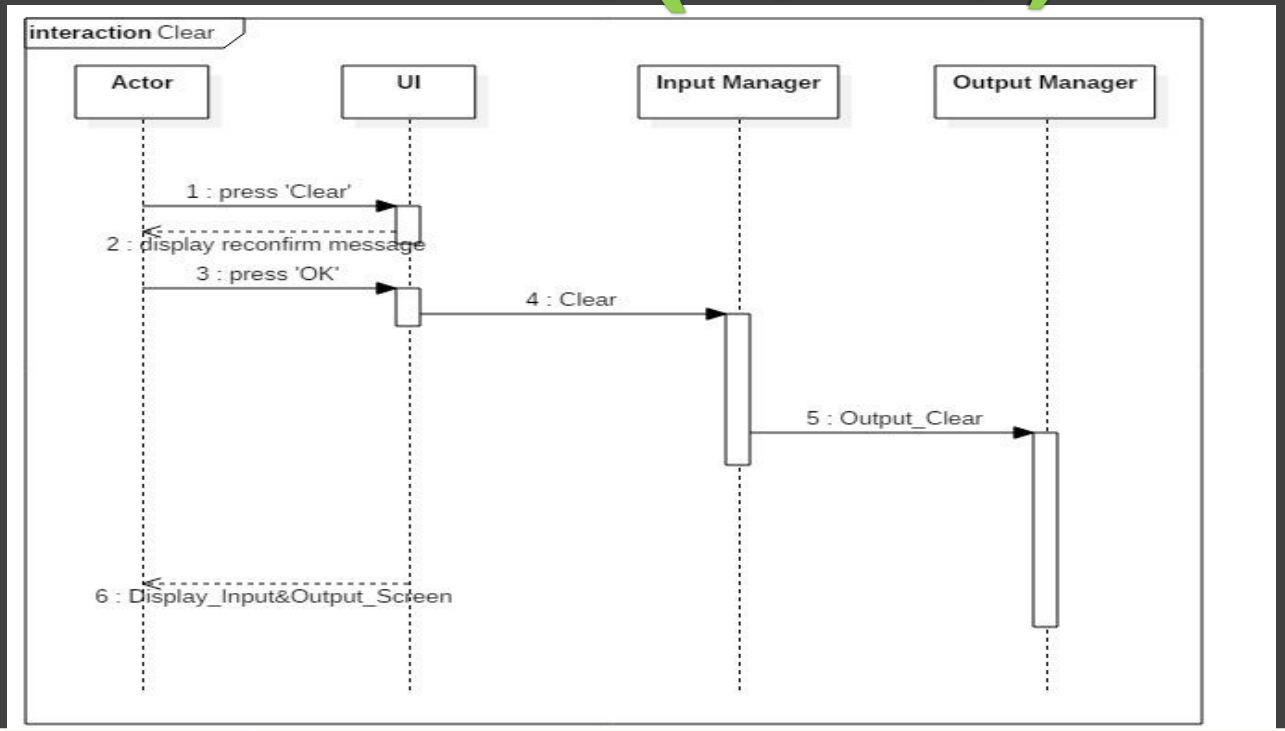
Name	Delete_Constraintbutton
Responsibilities	Constraint 삭제 버튼을 클릭한다.
Type	GUI
Cross Reference	R1.8
Notes	Constraint를 삭제해주는 Delete_Constraint메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Delete_Constraint()가 실행된다.

Implement Windows(12/13)



Name	CalButton
Responsibilities	Calculate 버튼을 클릭한다.
Type	GUI
Cross Reference	R2.1
Notes	TestCase를 계산해주는 Calculate()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Calculate()가 실행된다.

Implement Windows(13/13)



Name	ClearButton
Responsibilities	Clear 버튼을 클릭한다.
Type	GUI
Cross Reference	R3.1
Notes	초기화를 해주는 Clear()메서드를 호출한다.
Pre-Conditions	N/A
Post-Conditions	Clear()가 실행된다.

Unit Testing

Write Unit Test Code(1/4)

```java
    @Test
    public void testInput_Manager() {
        Input_Manager im = new Input_Manager();

        assertNotNull(im.Category_List);
        assertNotNull(im.Selected_RV_List);
        assertNotNull(im.Property_List);

    }


    @Test
    public void testAdd_Category() {
        Input_Manager im = new Input_Manager();

        im.Add_Category();
        im.Add_Category();
        im.Add_Category();

        assertEquals(3, im.Category_List.size());
        assertEquals(3, im.Category_Num);
    }

    @Test
    public void testChange_Category() {
        Input_Manager im = new Input_Manager();

        im.Add_Category();
        im.Change_Category(0, "1");

        assertEquals("1", im.Search_Category(0).Category_Name);
    }

    @Test
    public void testDelete_Category() {
        Input_Manager im = new Input_Manager();

        im.Add_Category();
        im.Add_Category();
        im.Delete_Category(0);

        assertEquals(1, im.Category_List.size());
        assertEquals(2, im.Category_Num);
        assertEquals(null,im.Search_Category(0));
    }

```

```java
    @Test
    public void testAdd_RV() {
        Input_Manager im = new Input_Manager();

        im.Add_Category();
        im.Add_RV(0);
        im.Add_RV(0);
        im.Add_RV(0);

        assertEquals(3, im.Category_List.get(0).RV_List.size());
        assertEquals(3, im.Category_List.get(0).RV_Num);

    }

    @Test
    public void testChange_RV() {
        Input_Manager im = new Input_Manager();

        im.Add_Category();
        im.Add_RV(0);
        im.Change_RV(0,0, "1");

        assertEquals("1", im.Search_Category(0).Search_RV(0).RV_Name);
    }

    @Test
    public void testDelete_RV() {
        Input_Manager im = new Input_Manager();

        im.Add_Category();
        im.Add_RV(0);
        im.Add_RV(0);
        im.Add_RV(0);
        im.Delete_RV(0,0);

        assertEquals(2, im.Search_Category(0).RV_List.size());
        assertEquals(3, im.Category_List.get(0).RV_Num);

        assertEquals(null,im.Search_Category(0).Search_RV(0));
    }

```

Write Unit Test Code(2/4)

```java
@Test
public void testAdd_Constraint() {
    Input_Manager im = new Input_Manager();

    im.Add_Category();
    im.Add_RV(0);
    im.Add_Constraint(0, 0, "Single", "");
    assertEquals(1, im.Search_Category(0).Search_RV(0).Single_Error);

    im.Add_Constraint(0, 0, "Error", "");
    assertEquals(2, im.Search_Category(0).Search_RV(0).Single_Error);
    im.Add_Constraint(0, 0, "Property", "1");
    im.Add_Constraint(0, 0, "Property", "1");
    assertEquals(1, im.Search_Category(0).Search_RV(0).RV_Property_List.size());
    im.Add_Constraint(0, 0, "If", "1");
    im.Add_Constraint(0, 0, "If", "1");
    im.Add_Constraint(0, 0, "If", "2");
    assertEquals(1, im.Search_Category(0).Search_RV(0).If_List.size());
}

@Test
public void testDelete_Constraint() {
    Input_Manager im = new Input_Manager();

    im.Add_Category();
    im.Add_RV(0);
    im.Add_Constraint(0, 0, "Single", "");
    im.Add_Constraint(0, 0, "Property", "1");
    im.Add_Constraint(0, 0, "If", "1");
    im.Delete_Constraint(0,0, "Single", "");
    im.Delete_Constraint(0,0,"Property", "1");
    im.Delete_Constraint(0,0,  "If", "1");
    assertEquals(0, im.Search_Category(0).Search_RV(0).Single_Error);
    assertEquals(0, im.Search_Category(0).Search_RV(0).RV_Property_List.size());
    assertEquals(0, im.Search_Category(0).Search_RV(0).If_List.size());
}
```

```java
@Test
public void testCalculate() {
    Input_Manager input_manager= new Input_Manager();

    input_manager.Add_Category();
    input_manager.Add_RV(0);
    input_manager.Add_Constraint(0,0,"Error","");
    input_manager.Add_RV(0);
    input_manager.Add_Constraint(0,1,"Error","");
    input_manager.Add_RV(0);
    input_manager.Select_RV(0,0);

    input_manager.Change_Category(0, "Model number");
    input_manager.Change_RV(0, 0,"Malformed");
    input_manager.Change_RV(0, 1,"Not in database");
    input_manager.Change_RV(0, 2,"Valid");

    input_manager.Add_Category();
    input_manager.Add_RV(1);
    input_manager.Add_Constraint(1,0,"Single","");
    input_manager.Add_RV(1);
    input_manager.Add_Constraint(1,1,"Single","");
    input_manager.Add_Constraint(1,1,"Property","RSNE");
    input_manager.Add_RV(1);
    input_manager.Add_Constraint(1,2,"Property","RSNE");
    input_manager.Add_Constraint(1,2,"Property","RSMANY");

    input_manager.Change_Category(1, "#SMRS");
    input_manager.Change_RV(1, 0,"0");
    input_manager.Change_RV(1, 1,"1");
    input_manager.Change_RV(1, 2,"MANY");

    input_manager.Add_Category();
    input_manager.Add_RV(2);
    input_manager.Add_Constraint(2,0,"Single","");
    input_manager.Add_RV(2);
    input_manager.Add_Constraint(2,1,"Single","");
    input_manager.Add_Constraint(2,1,"Property","OSNE");
    input_manager.Add_RV(2);
    input_manager.Add_Constraint(2,2,"Property","OSNE");
    input_manager.Add_Constraint(2,2,"Property","OSMANY");

    input_manager.Change_Category(2, "#SMOS");
    input_manager.Change_RV(2, 0,"0");
    input_manager.Change_RV(2, 1,"1");
    input_manager.Change_RV(2, 2,"MANY");
```

Write Unit Test Code(3/4)

```
196
197         input_manager.Add_Category();
198         input_manager.Add_RV(3);
199         input_manager.Add_Constraint(3,0,"Error","");
200         input_manager.Add_RV(3);
201         input_manager.Add_Constraint(3,1,"Single","");
202         input_manager.Add_RV(3);
203
204         input_manager.Change_Category(3, "#DBM");
205         input_manager.Change_RV(3, 0,"0");
206         input_manager.Change_RV(3, 1,"1");
207         input_manager.Change_RV(3, 2,"MANY");
208
209         input_manager.Add_Category();
210         input_manager.Add_RV(4);
211         input_manager.Add_Constraint(4,0,"Error","");
212         input_manager.Add_RV(4);
213         input_manager.Add_Constraint(4,1,"Single","");
214         input_manager.Add_RV(4);
215
216         input_manager.Change_Category(4, "#DBC");
217         input_manager.Change_RV(4, 0,"0");
218         input_manager.Change_RV(4, 1,"1");
219         input_manager.Change_RV(4, 2,"MANY");
220
221         input_manager.Add_Category();
222         input_manager.Add_RV(5);
223         input_manager.Add_Constraint(5,0,"Error","");
224         input_manager.Add_RV(5);
225         input_manager.Add_Constraint(5,1,"Error","");
226         input_manager.Add_RV(5);
227         input_manager.Add_Constraint(5,2,"Error","");
228         input_manager.Add_RV(5);
229
230         input_manager.Change_Category(5, "Correspondence of selection with model slots");
231         input_manager.Change_RV(5, 0,"Omitted slots");
232         input_manager.Change_RV(5, 1,"Extra slots");
233         input_manager.Change_RV(5, 2,"Mismatched slots ");
234         input_manager.Change_RV(5, 3,"Complete correspondence ");
235
236         input_manager.Add_Category();
237         input_manager.Add_RV(6);
238         input_manager.Add_Constraint(6,0,"Error","");
239         input_manager.Add_Constraint(6,0,"If","RSNE");
240         input_manager.Add_RV(6);
241         input_manager.Add_Constraint(6,1,"Error","");
```

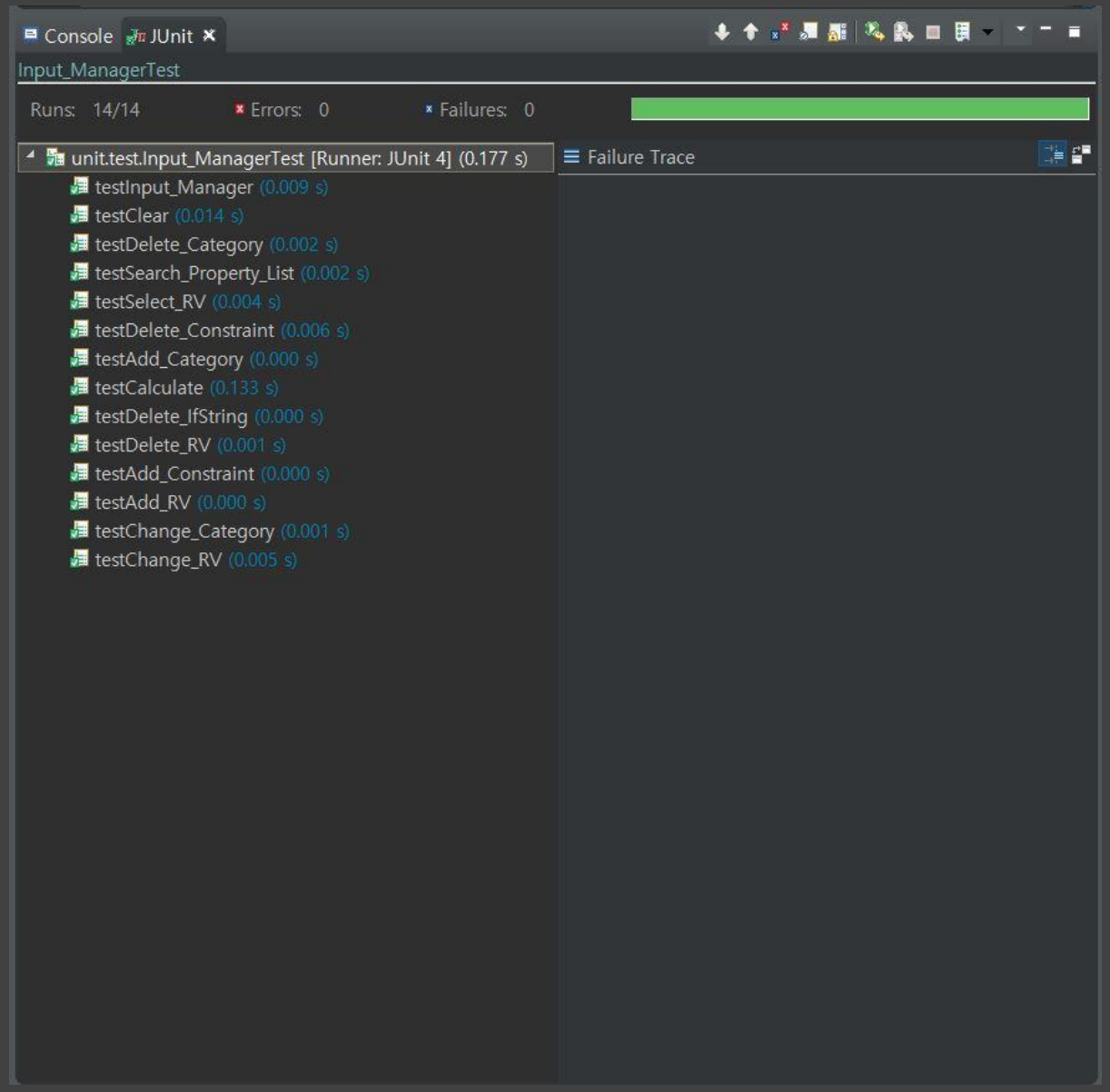
```
242         input_manager.Add_Constraint(6,1,"If","RSNE");
243         input_manager.Add_RV(6);
244         input_manager.Add_Constraint(6,2,"If","RSMANY");
245
246         input_manager.Change_Category(6, "of required components");
247         input_manager.Change_RV(6, 0,"0");
248         input_manager.Change_RV(6, 1,"<RS");
249         input_manager.Change_RV(6, 2,"=RS");
250
251         input_manager.Add_Category();
252         input_manager.Add_RV(7);
253         input_manager.Add_Constraint(7,0,"Single","");
254         input_manager.Add_RV(7);
255         input_manager.Add_RV(7);
256         input_manager.Add_RV(7);
257         input_manager.Add_RV(7);
258         input_manager.Add_RV(7);
259         input_manager.Add_Constraint(7,5,"Error","");
260
261         input_manager.Change_Category(7, "Required component selection");
262         input_manager.Change_RV(7, 0,"Some defaults");
263         input_manager.Change_RV(7, 1,"All valid");
264         input_manager.Change_RV(7, 2,">=1 incompatible with slots");
265         input_manager.Change_RV(7, 3,">=1 incompatible with another selection");
266         input_manager.Change_RV(7, 4,">=1 incompatible with model");
267         input_manager.Change_RV(7, 5,">=1 not in database ");
268
269         input_manager.Add_Category();
270         input_manager.Add_RV(8);
271         input_manager.Add_RV(8);
272         input_manager.Add_Constraint(8,1,"If","OSNE");
273         input_manager.Add_RV(8);
274         input_manager.Add_Constraint(8,2,"If","OSMANY");
275
276         input_manager.Change_Category(8, "# of optional components");
277         input_manager.Change_RV(8, 0,"< #SMOS");
278         input_manager.Change_RV(8, 1,"= #SMOS");
279
280         input_manager.Add_Category();
281         input_manager.Add_RV(9);
282         input_manager.Add_Constraint(9,0,"Single","");
283         input_manager.Add_RV(9);
284         input_manager.Add_RV(9);
285         input_manager.Add_RV(9);
286         input_manager.Add_RV(9);
287         input_manager.Add_RV(9);
```

Write Unit Test Code(4/4)

```
287         input_manager.Add_RV(9);
288         input_manager.Add_Constraint(9,5,"Error","");
289 
290         input_manager.Change_Category(9, "Optional component selection");
291         input_manager.Change_RV(7, 0,"Some defaults");
292         input_manager.Change_RV(7, 1,"All valid");
293         input_manager.Change_RV(7, 2,">=1 incompatible with slots");
294         input_manager.Change_RV(7, 3,">=1 incompatible with another selection");
295         input_manager.Change_RV(7, 4,">=1 incompatible with model");
296         input_manager.Change_RV(7, 5,">=1 not in database ");
297 
298         input_manager.Calculate();
299         assertEquals(input_manager.cal.All_TC.size(),67);
300         assertEquals(input_manager.cal.Selected_TC.size(),1);
301     }
302 
303     @Test
304     public void testClear() {
305         Input_Manager im = new Input_Manager();
306 
307         im.Add_Category();
308         im.Add_RV(0);
309         im.Add_RV(0);
310         im.Add_RV(0);
311 
312         im.Clear();
313 
314         assertEquals(im.Category_List.size(),0);
315         assertEquals(im.cal.All_TC.size(),0);
316         assertEquals(im.cal.Selected_TC.size(),0);
317 
318 
319     }
320 
```

```
322     @Test
323     public void testDelete_IfString() {
324         Input_Manager im = new Input_Manager();
325         im.Add_Category();
326         im.Add_RV(0);
327         im.Add_RV(0);
328         im.Add_Constraint(0, 0, "Single", "");
329         im.Add_Constraint(0, 0, "Property", "1");
330         im.Add_Constraint(0, 0, "If", "1");
331         im.Add_Constraint(0, 1, "If", "1");
332         im.Delete_If("1");
333         assertEquals(im.Search_Category(0).Search_RV(0).If_List.size(),0);
334         assertEquals(im.Search_Category(0).Search_RV(1).If_List.size(),0);
335     }
336 
337     @Test
338     public void testSearch_Property_List() {
339         Input_Manager im = new Input_Manager();
340         im.Add_Property_List("1");
341         assertEquals(true,im.Search_Property_List("1"));
342         assertEquals(false,im.Search_Property_List("2"));
343 
344     }
345 
346 
347 }
348 
```

Unit Testing



System Testing

System Testing(1/3)

Test Number	Test 항목	Description	Use Case	System Function	Pass/Fail
1-1	Category 추가 Test	Category를 추가 할 때 Category #number로 추가되는지 확인한다.	Category 추가	R1.1	P
2-1	Category 수정 Test	Category의 이름이 수정되는지 확인한다.	Category 수정	R1.2	P
2-2	Category 수정 Test	Category의 수정을 요청하고 이름을 입력하지 않았을 때 기존의 이름인지 확인한다.	Category 수정	R1.2	P
3-1	Category 삭제 Test	Category의 삭제를 요청하고 승인하면 Category가 삭제되는지 확인한다.	Category 삭제	R1.3	P
3-2	Category 삭제 Test	Category의 삭제를 요청하고 승인하지 않으면 Category가 유지되는지 확인한다.	Category 삭제	R1.3	P
3-3	Category 삭제 Test	Category내에 Property가 존재할 때 삭제가 안되는지 확인한다.	Category 삭제	R1.3	P
4-1	대푯값 추가 Test	대푯값을 추가 할 때 RV #number로 추가되는지 확인한다.	대푯값 추가	R1.4	P

System Testing(2/3)

Test Number	Test 항목	Description	Use Case	System Function	Pass/Fail
5-1	대푯값 수정 Test	대푯값의 이름이 수정되는지 확인한다 .	대푯값 수정	R1.5	P
5-2	대푯값 수정 Test	대푯값의 수정을 요청하고 이름을 입력하지 않았을 때 기존의 이름인지 확인한다.	대푯값 수정	R1.5	P
6-1	대푯값 삭제 Test	대푯값의 삭제를 요청하고 승인하면 대푯값이 삭제되는지 확인한다.	대푯값 삭제	R1.6	P
6-2	대푯값 삭제 Test	대푯값의 삭제를 요청하고 승인하지 않으면 대푯값 유지되는지 확인한다.	대푯값 삭제	R1.6	P
6-3	대푯값 삭제 Test	대푯값을 삭제할때 Propety가 존재하면 관련된 If를 삭제하는지 확인한다.	대푯값 삭제	R1.6	P
7-1	Constraint 추가 Test	Constraint의 이름과 속성을 입력하고 추가 할 때 입력된 이름과 속성대로 추가되는지 확인한다.	Constraint 추가	R1.7	P
7-2	Constraint 추가 Test	Constraint 추가를 위해 선택한 RV가 표시가 되는지 확인한다.	Constraint 추가	R1.7	P

System Testing(3/3)

Test Number	Test 항목	Description	Use Case	System Function	Pass/Fail
7-3	Constraint 추가 Test	Property 속성을 갖는 Constraint가 존재하지 않을 때, If Property 속성을 갖는 Constraint 추가요청이 들어올 경우 Constraint를 추가하지 않는지 확인한다.	Constraint 추가	R1.7	P
8-1	Constraint 삭제 Test	Constraint의 삭제를 요청하고 승인하면 Constraint가 삭제되는지 확인한다.	Constraint 삭제	R1.8	P
8-2	Constraint 삭제 Test	Property를 삭제하면 관련된 If가 모두 삭제되는지 확인한다.	Constraint 삭제	R1.8	P
9-1	대푯값 선택 Test	대푯값을 선택하고 Test Case를 계산 했을 때 화면에 선택한 대푯값의 Test Case가 출력되는지 확인한다.	대푯값 선택	R1.9	P
10-1	Test Case 계산 Test	Test Case를 예상 결과값과 비교한다.	Test Case 계산	R2.1	F
11-1	초기화 Test	초기화 요청 시 모든 입력과 출력이 삭제되는지 확인한다.	초기화	R3.1	P

Demonstration