“Priority” Category Partitioning Testing Tool

OOPT stage 2030

Analyze

Project Team

T4

201311265 김상원

201210194 김정환

201311269 김제헌

201311297 이상명

Date

2017-05-31

목차

[Activity2010. Revise Plan 3](#_Toc484013121)

[Activity2020. Synchronize Artifacts 3](#_Toc484013122)

[Activity2031. Define Essential Use Case 3](#_Toc484013123)

[Activity2032. Refine Use Case Diagrams 10](#_Toc484013124)

[Activity2033. Define Domain Model 11](#_Toc484013125)

[Activity2034. Refine Glossary 11](#_Toc484013126)

[Activity2035. Define System Sequence Diagrams 14](#_Toc484013127)

[Activity2036. Define Operation Contracts 17](#_Toc484013128)

[Activity2038. Refine System Test Case 24](#_Toc484013129)

[Activity2039. Traceability Analysis 26](#_Toc484013130)

# Activity2010. Revise Plan

OOPT Stage 1000 <Plan and Elaboration> 보고서 Version. 3.0 수정

# Activity2020. Synchronize Artifacts

OOPT Stage 1000 <Plan and Elaboration> 보고서 Version 3.0 수정

# Activity2031. Define Essential Use Case

|  |  |
| --- | --- |
| **Use Case** | Make new specification file |
| **Actor** | User |
| **Purpose** | Test case generate을 위해 새로운 specification 파일을 작성한다. |
| **Overview** | 초기화면에서 ‘새로 만들기’버튼을 클릭하면 저장할 파일 이름을 입력 받는 화면이 출력되어, 입력 받은 이름으로 새로운 specification파일을 생성한다. 최근 파일 항목에 반영한다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R1.1 |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor, (S):System  1. (A) 초기화면에 있는 ‘새로 만들기’ 버튼을 누른다.  2. (S) specification 파일 이름을 입력 받는 단계로 넘어간다.  3. (A) 새 파일 이름 항목란에 이름을 입력한다.  4. (A) ‘이름 결정’ 버튼을 누른다.  5. (S) 지정된 이름의 specification 파일을 만든다.  6. (S) 생성된 specification 파일을 recent file list에 갱신한다.  7. (S) Set category page로 이동한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | E. 입력한 이름과 같은 파일이 존재하면 예외처리한다. |

|  |  |
| --- | --- |
| **Use Case** | Load specification file |
| **Actor** | User |
| **Purpose** | 이미 작성했던 specification중 저장한 파일을 불러온다. |
| **Overview** | 초기화면에서 ‘불러오기’버튼을 클릭하거나, 최근 파일 목록 중 하나를 선택하여 기존 작성한 명세를 불러온다. 최근 파일 항목에 반영한다 |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R1.2 |
| **Pre-Requisites** | 불러올 specification파일 또는 recent file list에 specification  파일이 있어야 한다. |
| **Typical Courses of Events** | (A) : Actor, (S):System  1.1 (A) 초기화면에 있는 ‘불러오기’ 버튼을 누른다.  1.2 (S) specification 파일을 찾기 위한 탐색기 창으로 넘어간다.  1.3. (A) 탐색기 창에서 specification 파일을 선택한다.  1.4. (S) 선택된 specification 파일을 불러 읽는다.  1.5. (S) 선택된 specification 파일을 recent file list에 갱신한다.  1.6. (S) specification 파일이 저장된 시점의 값들을 보여준다.  2.1. (A) 초기화면에 있는 recent file list에서 한 항목을 더블  클릭한다.  2.2. (S) 선택된 specification 파일을 불러 읽는다.  2.3. (S) 선택된 specification 파일의 recent file list 순위를  갱신한다.  2.4. (S) specification 파일이 저장된 시점의 값들을 보여준다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | E. 최근 파일 항목에 있던 파일이 위치가 이동되거나  삭제되어 주소가 변경되었을 때, 오류 메시지를 호출한다. |

|  |  |
| --- | --- |
| **Use Case** | Shut down program |
| **Actor** | User |
| **Purpose** | 프로그램 종료 |
| **Overview** | 초기화면에서 종료하기 버튼을 클릭하여 프로그램을 종료한다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R1.3 |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor, (S):System  1. (A) 초기화면에 있는 ‘종료하기’ 버튼을 누른다.  2. (S) 프로그램을 종료한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | N/A |

|  |  |
| --- | --- |
| **Use Case** | Set category |
| **Actor** | User |
| **Purpose** | Specification의 각 category를 편집한다. |
| **Overview** | 새로운 category를 추가 또는 이미 작성된 category를 삭제 또는 이미 작성된 category를 수정한다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R2.1, R2.2, R2.3, R2.4 |
| **Pre-Requisites** | ‘수정’, ‘삭제’버튼이 작동하기 위해서는 이미 추가된 category가 있어야 한다. |
| **Typical Courses of Events** | (A) : Actor, (S):System  1.1. (A) Category 목록 밑에 있는 추가 버튼을 누른다.  1.2. (S) category 항목 란을 추가한다.  1.2. (A) 추가된 category 항목 란에 category 이름을 입력한다.  1.3. (S) category 이름들을 specification에 반영한다.  2.1. (A) Category 목록 밑에 있는 삭제 버튼을 누른다.  2.2. (S) 선택된 category 항목과 하위 representative values항을 삭제한다.  2.3. (S) 삭제된 category 값과 하위 representative values값들을 specification에 반영한다.  3.1. (A) Category 목록 밑에 있는 수정 버튼을 누른다.  3.2. (A) 선택된 category 항목의 새로운 category 이름을 입력한다.  3.3. (S) 선택된 category 항목의 이름을 새로운 이름으로 변경한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | E. 같은 category 이름이 존재하면 예외처리한다. |

|  |  |
| --- | --- |
| **Use Case** | Set representative values |
| **Actor** | User |
| **Purpose** | Specification의 각 representative value를 편집한다. |
| **Overview** | 새로운 representative value 를 추가 또는 이미 작성된 representative value 를 삭제 또는 이미 작성된 representative value를 수정한다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R2.2, R2.3, R2.4 |
| **Pre-Requisites** | ‘수정’, ‘삭제’버튼이 작동하기 위해서는 이미 추가된 representativeValue가 있어야 한다. |
| **Typical Courses of Events** | (A) : Actor, (S):System  1.1. (A) Category 항목 안에 있는 추가 버튼을 누른다.  1.2. (S) category 항목 안에 representative value 항목 란을  추가한다.  1.2. (A) 추가된 representative value 항목 란에  representative value 이름을 입력한다.  1.3. (S) representative value 이름들을 specification에 반영한다.  2.1. (A) Category 항목 안에 있는 삭제 버튼을 누른다.  2.2. (S) 선택된 representative value 항목 란을 삭제한다.  2.3. (S) 삭제된 representative values값을 specification에  반영한다.  3.1. (A) Category 항목 안의 representative value 항목 수정 버튼을 누른다.  3.2. (A) 선택된 representative value 항목의 새로운 representative value 이름을 입력한다.  3.3. (S) 선택된 representative value 항목의 이름을 새로운 이름으로 변경한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | E. 같은 representative value 이름이 존재하면 예외처리한다. |

|  |  |
| --- | --- |
| **Use Case** | Set property |
| **Actor** | User |
| **Purpose** | Specification의 각 property constraint를 설정한다. |
| **Overview** | 각 representative value마다 property constraint를 부여하거나 삭제 또는 수정할 수 있다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R2.3, R2.4 |
| **Pre-Requisites** | ‘수정’, ‘삭제’버튼이 작동하기 위해서는 이미 추가된 property가 있어야 한다. |
| **Typical Courses of Events** | (A) : Actor, (S):System  1.1. (A) representative value의 property constraint 항목의 추가 버튼을 누른다.  1.2. (A) 새로운 property constraint 항목의 이름을 입력한다.  1.3. (S) property constraint를 추가한다.  2.1. (A) representative value의 property constraint 항목의 삭제 버튼을 누른다.  2.2. (S) property constraint 항목을 삭제한다.  3.1. (A) representative value의 property constraint 항목의 수정 버튼을 누른다.  3.2. (A) 새로운 property constraint 이름을 입력한다.  3.3. (S) property constraint 항목의 이름을 새로운 이름으로 변경한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | E. 같은 property 이름이 존재하면 예외처리한다. |

|  |  |
| --- | --- |
| **Use Case** | Set if-property |
| **Actor** | User |
| **Purpose** | Specification의 각 if-property constraint를 설정한다. |
| **Overview** | 각 representative value마다 기존에 설정된 property constraints 중에서 if-property constraint를 추가 또는 삭제 또는 수정할 수 있다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R2.4 |
| **Pre-Requisites** | ‘수정’, ‘삭제’버튼이 작동하기 위해서는 이미 추가된 if-property가 있어야 한다. |
| **Typical Courses of Events** | (A) : Actor, (S):System  1.1. (A) representative value의 if-property constraint 항목의 추가 버튼을 누른다.  1.2. (A) 기존에 설정된 property constraints 중에서 하나의property constraint 항목을 선택한다.  1.3. (S) 선택된 property constraint를 추가한다.  2.1. (A) representative value의 기존 if-property constraint 항목을 선택한 후 삭제 버튼을 누른다.  2.2. (S) 선택된 if-property constraint를 삭제한다.  3.1. (A) representative value의 기존 if-property constraint 항목을 선택한 후 수정 버튼을 누른다.  3.2. (A) 기존 if-property constraint 항목들 중에서 하나의 property constraint 항목을 선택한다.  3.3. (S) 기존의 if-property constraint 항목을 새롭게 선택된 항목으로 변경한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | N/A |

|  |  |
| --- | --- |
| **Use Case** | Set single and error |
| **Actor** | User |
| **Purpose** | Specification의 각 single 또는 error constraint를 설정한다. |
| **Overview** | 각 representative value마다 single 또는 error constraint를  부여하거나 지울 수 있다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements  : R2.5 |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor, (S):System  1. (A) representative value의 constraint를 single 또는 error  constraints 중에서 선택한다.  2. (S) single 또는 error constraint들을 specification에 반영한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | N/A |

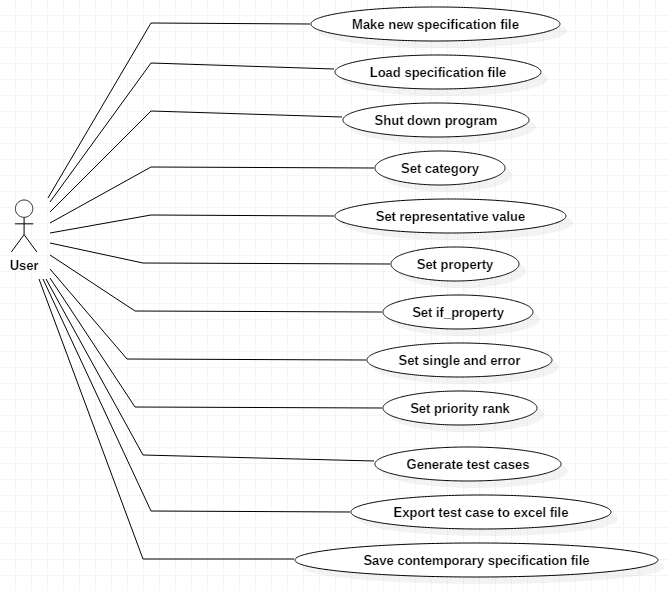
|  |  |
| --- | --- |
| **Use Case** | Set priority rank |
| **Actor** | User |
| **Purpose** | Specification의 각 priority를 설정한다. |
| **Overview** | 입력된 representative value들에게 사용자가 priority(중요도)를 부여한다.  입력할 수 있는 priority(중요도)는 총 5단계[1~5점]로 정해진다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R2.6 |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor, (S):System  1. (A) representative value의 priority를 1, 2, 3, 4, 5 중에서  선택한다.  2. (S) priority들을 specification에 반영한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | N/A |

|  |  |
| --- | --- |
| **Use Case** | Generate test cases |
| **Actor** | User |
| **Purpose** | Specification이 확인되면 test case generate가 시작된다. |
| **Overview** | 입력된 specification을 통해 test case를 만들고 그 총 개수를 화면을 통해 사용자에게 알려준다.  사용자는 생성된 test cases의 수를 보고 generate이 잘 되었는지 판단할 수 있다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R3 |
| **Pre-Requisites** | 입력된 Specification이 test case를 만들기 위한 충분한 값이 입력되어야 한다. |
| **Typical Courses of Events** | (A) : Actor, (S):System  1. (A) ‘Test case 생성 후 엑셀로 저장하기’ 버튼을 누른다.  2. (S) 입력된 specification을 통해 test case를 만든다.  3. (S) 총 개수를 새로운 화면을 통해 사용자에게 알려준다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | 입력된 Specification이 test case를 만들기 위한 충분한 값이 입력되어야 한다. |

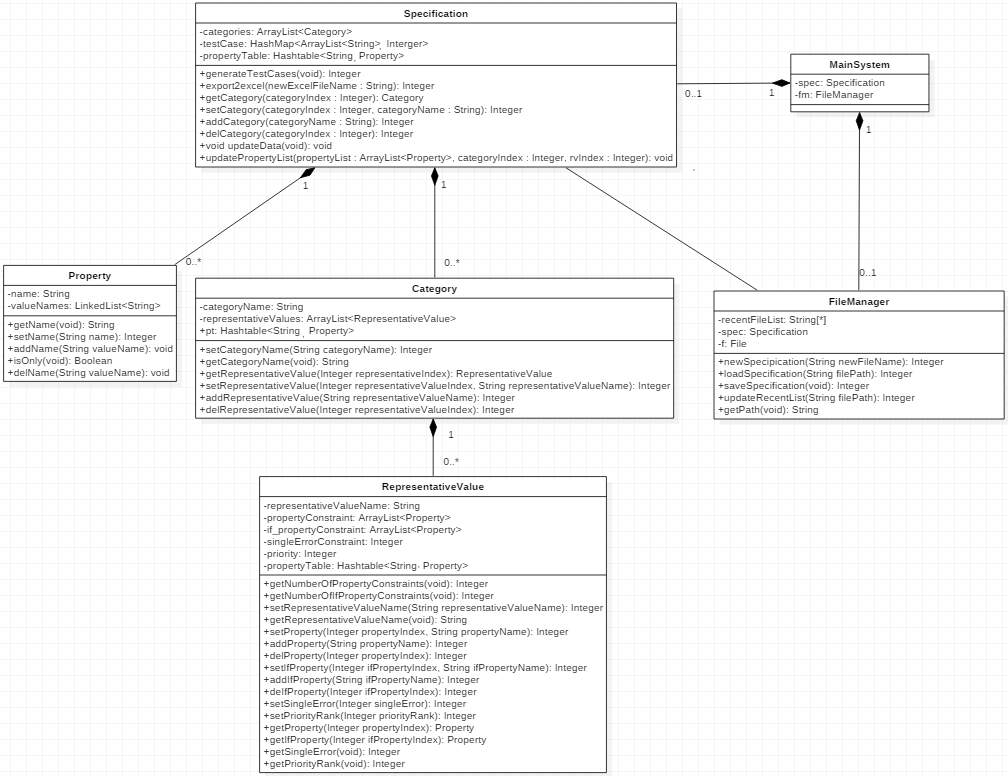
|  |  |
| --- | --- |
| **Use Case** | Export test case to excel file |
| **Actor** | User |
| **Purpose** | Test cases를 엑셀 파일로 저장한다. |
| **Overview** | Test case generate이 끝난 후, 생성할 엑셀파일의 이름을 받는다. 이후 생성된 test cases를 엑셀 파일로 저장한다.  생성된 엑셀파일은 설정한 Priority순으로 정렬 된다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R4 |
| **Pre-Requisites** | 입력된 Specification이 test case를 만들기 위한 충분한 값이 입력되어서, Generate test cases가 실행되어야 한다. |
| **Typical Courses of Events** | (A) : Actor, (S):System  1. (S) ‘Generate test cases가 실행된다.  2. (S) 생성할 엑셀파일의 이름을 받는 창이 뜬다.  3. (A) 생성할 엑셀파일의 이름을 입력한다.  4. (S) 생성된 test cases를 Priority순으로 정렬 한다.  5. (S) 입력받은 엑셀파일의 이름으로 엑셀 파일을 만든 후,  Priority순으로 정렬된 test cases를 저장한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** |  |

|  |  |
| --- | --- |
| **Use Case** | Save contemporary specification file |
| **Actor** | User |
| **Purpose** | 현재까지의 작업 상태를 저장한다. |
| **Overview** | specification입력 중 중간저장 버튼을 누르면, 현재까지의 작업 상태를 저장한다.  저장한 후 Load specification을 통해 언제든 불러올 수 있다. |
| **Type** | Primary |
| **Cross Reference** | Funtional Requirements : R5 |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor, (S):System  1. (A) 저장 요청을 보낸다.  2. (S) 현재까지의 작업 상태를 specification 파일로 저장한다. |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | N/A |

# Activity2032. Refine Use Case Diagrams



# Activity2033. Define Domain Model

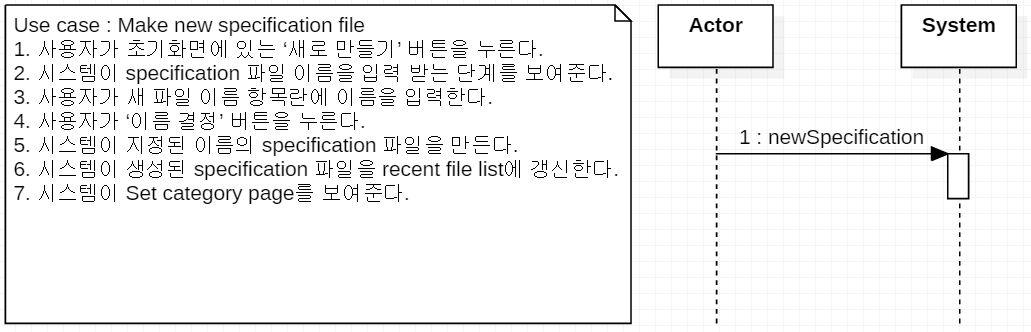


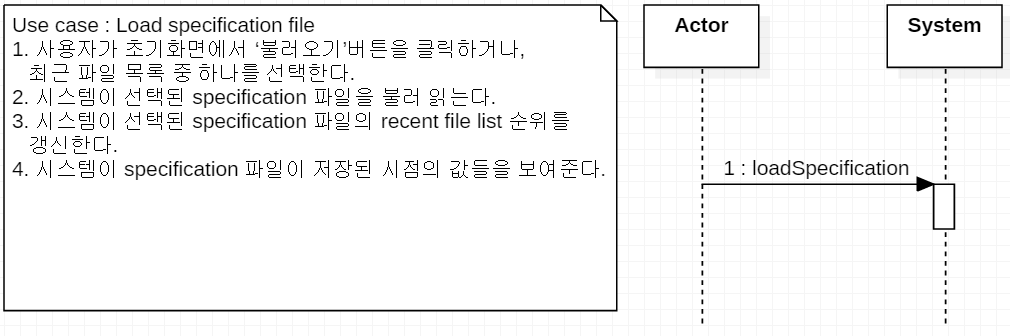
# Activity2034. Refine Glossary

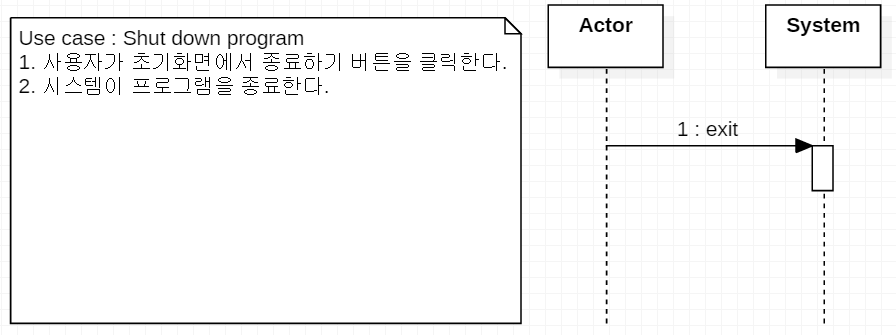
|  |  |  |
| --- | --- | --- |
| **Term.** | **Category** | **Description** |
| MainSystem | Class | 전반적인 프로그램 실행을 제어하는 클래스 |
| Specification | Class | 사용자가 CPT를 위해서 작성하는 Specification정보를 가지고 있는 클래스 |
| Category | Class | CPT를 위해 사용자가 설정한 Category를 나타내는 클래스 |
| RepresentativeValue | Class | Category에서 나타날 수 있는 대표값을 표현한 클래스 |
| FileManager | Class | File 입출력을 담당하는 클래스 |
| Property | Class | Property, If-property를 제어하기 위해 필요한 클래스 |
| Specification.generateTestCases | Operation | 입력된 specification으로부터 test case를 생성하는 메소드 |
| Specification .export2excel | Operation | 생성된 test case를 priority에 따라 정렬한 뒤 엑셀파일로 내보내는 메소드 |
| Specification.setCategory | Operation | Category를 변경하는 메소드 |
| Specification.getCategory | Operation | Category를 반환하는 메소드 |
| Specification.addCategory | Operation | Category를 추가하는 메소드 |
| Specification.delCategory | Operation | Category를 삭제하는 메소드 |
| Specification.updateData | Operation | Specification의 Data를 갱신하는 메소드 |
| Specification.updatePropertyList | Operation | PropertyList를 갱신하는 메소드 |
| Category.setCategoryName | Operation | Category의 이름을 변경하는 메소드 |
| Category.getCategoryName | Operation | Category의 이름을 반환하는 메소드 |
| Category.setRepresentativeValue | Operation | RepresentativeValue를 변경하는 메소드 |
| Category.getRepresentativeValue | Operation | RepresentativeValue를 반환하는 메소드 |
| Category.addRepresentativeValue | Operation | RepresentativeValue를 추가하는 메소드 |
| Category.delRepresentativeValue | Operation | RepresentativeValue를 삭제하는 메소드 |
| RepresentativeValue.setRepresentativeValueName() | Operation | RepresentativeValue의 이름을 변경하는 메소드 |
| RepresentativeValue.getRepresentativeValueName() | Operation | RepresentativeValue의 이름을 반환하는 메소드 |
| RepresentativeValue.setProperty | Operation | Property를 변경하는 메소드 |
| RepresentativeValue.getProperty | Operation | Property를 반환하는 메소드 |
| RepresentativeValue.addProperty | Operation | Property를 추가하는 메소드 |
| RepresentativeValue.delProperty | Operation | Property를 삭제하는 메소드 |
| RepresentativeValue.setIfProperty | Operation | IfProperty를 변경하는 메소드 |
| RepresentativeValue.getIfProperty | Operation | IfProperty를 반환하는 메소드 |
| RepresentativeValue.addIfProperty | Operation | IfProperty를 추가하는 메소드 |
| RepresentativeValue.delIfProperty | Operation | IfProperty를 삭제하는 메소드 |
| RepresentativeValue.setSingleError | Operation | Single, error constraints를 설정하는 메소드 |
| RepresentativeValue.getSingleError | Operation | 설정된 Single, error constraints를 반환하는 메소드 |
| RepresentativeValue.setPriorityRank | Operation | 중요도를 설정하는 메소드 |
| RepresentativeValue.getPriorityRank | Operation | 설정된 중요도를 반환하는 메소드 |
| FileManager.newSpecification | Operation | 새로운 명세 파일을 생성하는 메소드 |
| FileManager.loadSpecification | Operation | 기존 및 최근 명세 파일을 불러오는 메소드 |
| FileManager.saveSpecification | Operation | 현재까지 작업된 명세 파일을 저장하는 메소드 |
| FileManager.updateRecentList | Operation | 최근 파일 목록을 관리하는 메소드 |
| FileManager.getPath | Operation | 파일의 절대경로주소를 반환하는 메소드 |
| Property.getName | Operation | Property의 이름을 반환하는 메소드 |
| Property.setName | Operation | Property의 이름을 변경하는 메소드 |
| Property.addName | Operation | Property를 추가하는 메소드 |
| Property.isOnly | Operation | Property가 유일한지 확인하는 메소드 |
| Property.delName | Operation | Property를 삭제하는 메소드 |
| MainSystem.spec | Attribute | 사용자가 작성할 명세파일이 저장되는 변수 |
| MainSystem.fm | Attribute | 파일매니저를 이용하는 변수 |
| Specification.categories | Attribute | 명세정보의 카테고리를 저장하는 변수 |
| Sepcification.testCase | Attribute | 생성된 테스트를 저장하는 변수 |
| Sepcification.propertyTable | Attribute | Property들을 저장하는 변수 |
| Category. categoryName | Attribute | 카테고리의 이름을 저장하는 변수 |
| Category.representativeValues | Attribute | 카테고리의 대표값을 저장하는 변수 |
| Category.pt |  | Property와 카테고리 이름을 저장하는 변수 |
| RepresentativeValue. representativeValueName | Attribute | 대표값의 이름을 저장하는 변수 |
| RepresentativeValue.propertyConstraint | Attribute | 대표값의 property 제약조건을 저장하는 변수 |
| RepresentativeValue.if\_propertyConstraint | Attribute | 대표값의 if-property 제약조건을 저장하는 변수 |
| RepresentativeValue.singleErrorConstraint | Attribute | 대표값의 single, error 제약조건을 저장하는 변수 |
| RepresentativeValue.priority | Attribute | 대표값의 중요도를 저장하는 변수 |
| RepresentativeValue.propertyTable | Attribute | 대표값의 property들을 저장하는 변수 |
| FileManager.recentFileList | Attribute | 최근 파일 목록을 저장하는 변수 |
| FileManager.spec | Attribute | 사용자가 작성할 명세파일이 저장되는 변수 |
| FileManager.f | Attribute | 파일 객체를 저장하는 변수 |
| Property.name | Attribute | Property의 이름을 저장하는 변수 |
| Property.valueNames | Attribute | Property가 어디에 사용되고 있는지를 모두 저장하는 변수 |

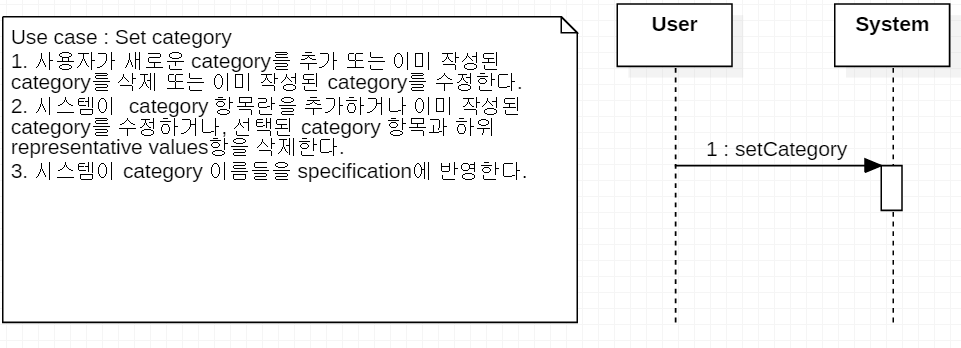
# Activity2035. Define System Sequence Diagrams

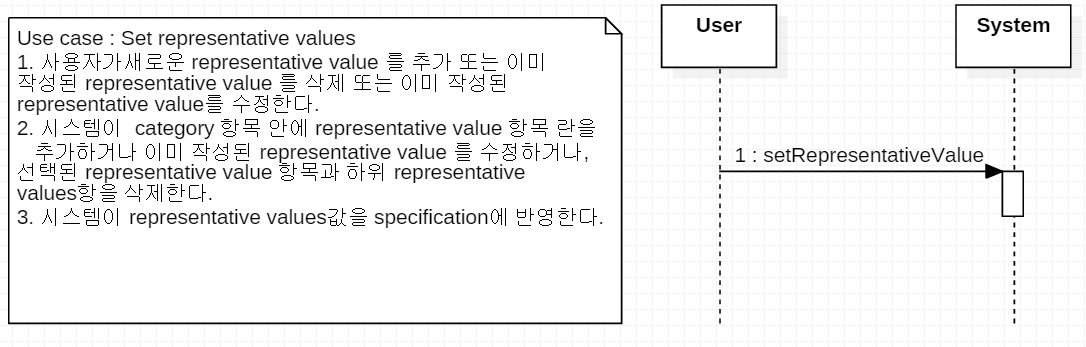
|  |  |
| --- | --- |
| **Use Case** | **Name of Actor-Activated Event** |
| 1.New specification file | newSpecification |
| 2.Load specification file | loadSpecification |
| 3.Shut down program | Exit |
| 4.Set category | setCategory |
| 5.Set representative value | setRepresentativeValue |
| 6.Set property | setProperty |
| 7.Set if-property | setIfProperty |
| 8.Set single and error | setSingleError |
| 9.Set priority rank | setPriorityRank |
| 10.Generate test cases | generateTestCases |
| 11.Export test case to excel file | export2excel |
| 12.Save contemporary specification file | saveSpecification |

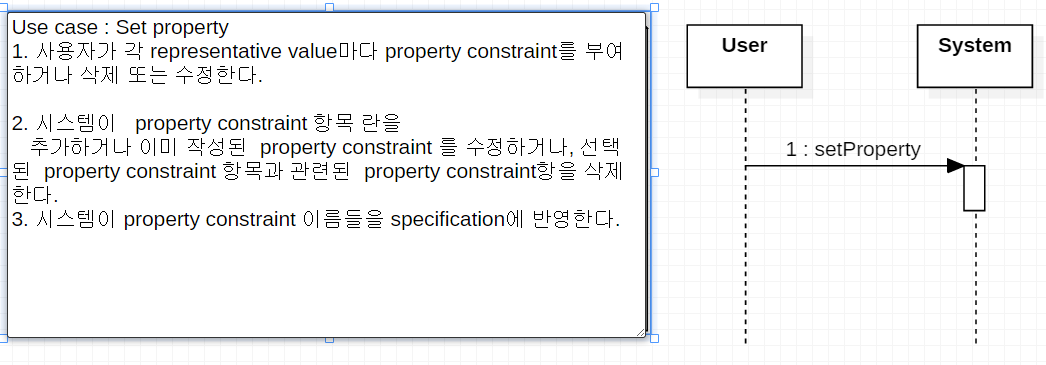


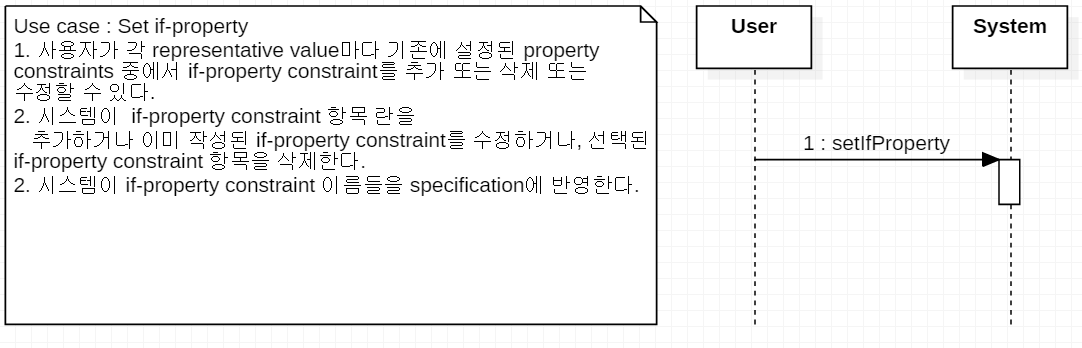


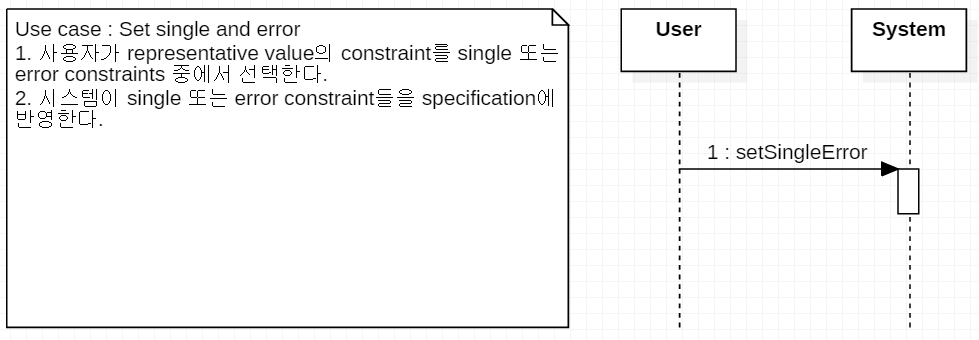


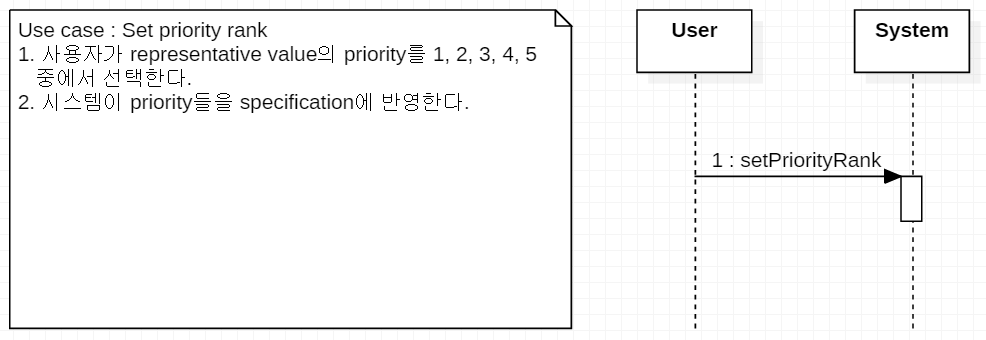


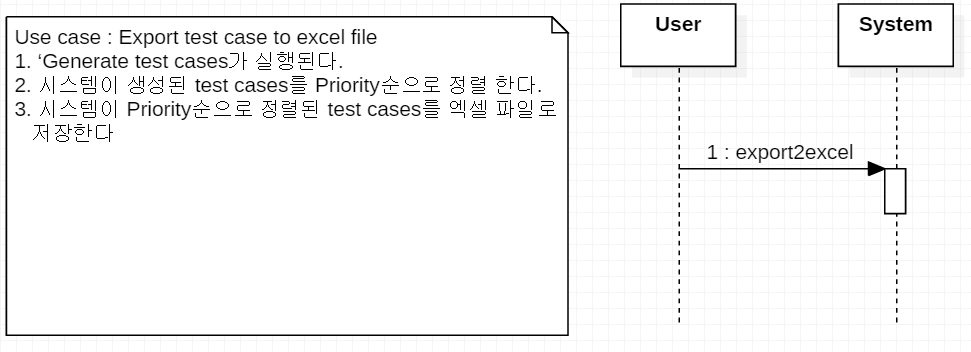
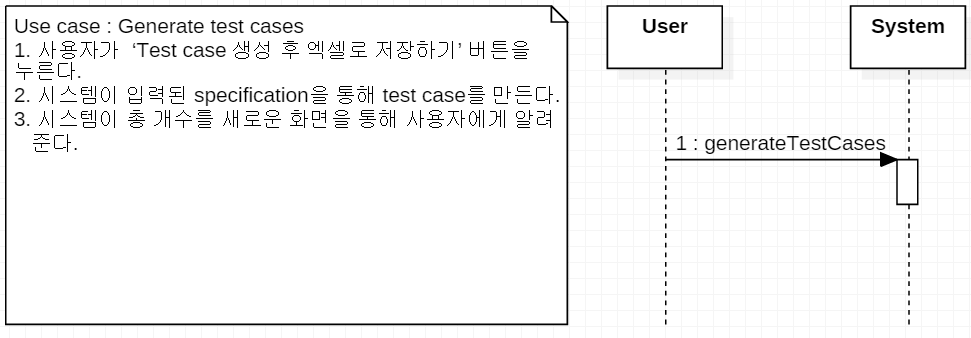


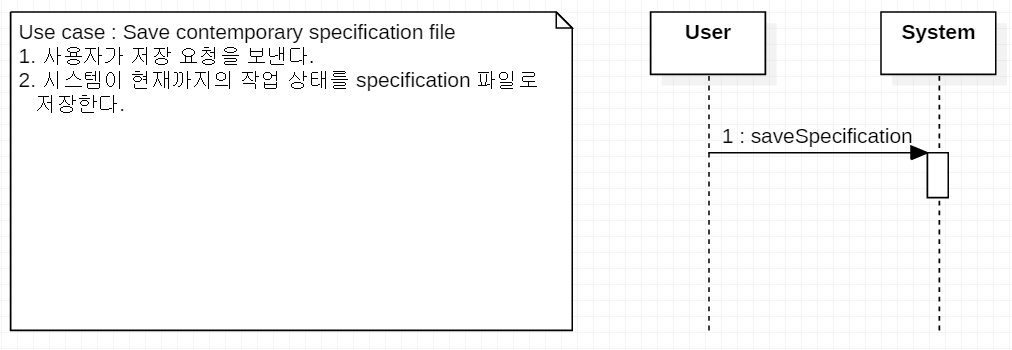












# Activity2036. Define Operation Contracts

|  |  |  |
| --- | --- | --- |
| **Use Case** | **Name of Actor-Activated Event** | **System Operation** |
| 1.New specification file | newSpecification | 1. newSpecification() |
| 2.Load specification file | loadSpecification | 2. loadSpecification() |
| 3.Shut down program | exit | 3. exit() |
| 4.Set category | setCategory | 4. addCategory() |
| 5. delCategory() |
| 6. setCategory() |
| 5.Set representative value | setRepresentativeValue | 7. addRepresentativeValue() |
| 8. delRepresentativeValue() |
| 9. setRepresentativeValue() |
| 6.Set property | setProperty | 10. addProperty() |
| 11. delProperty() |
| 12. setProperty() |
| 7.Set if-property | setIfProperty | 13. addIfProperty() |
| 14. delIfProperty() |
| 15. setIfProperty() |
| 8.Set single and error | setSingleError | 16. setSingleError() |
| 9.Set priority rank | setPriorityRank | 17. setPriorityRank() |
| 10.Generate test cases | generateTestCases | 18. generateTestCases() |
| 11.Export test case to excel file | export2excel | 19. export2excel() |
| 12.Save contemporary  specification file | saveSpecification | 20. saveSpecification() |

|  |  |
| --- | --- |
| **Name** | newSpecification() |
| **Responsibilities** | 새로운 명세파일을 생성한다. |
| **Type** | System |
| **Cross References** | R1.1 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 실제 새로운 파일이 생성되며, 카테고리 설정 화면으로 넘어간다. |

|  |  |
| --- | --- |
| **Name** | loadSpecification() |
| **Responsibilities** | 이미 작성하여 저장된 명세 파일을 불러온다. |
| **Type** | System |
| **Cross References** | R1.2 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | 컴퓨터 내에 불러올 명세 파일이 존재해야 한다. |
| **Post­conditions** | 저장된 명세가 로딩 되며, 저장 시 작업 단계로 이동된다. |

|  |  |
| --- | --- |
| **Name** | exit() |
| **Responsibilities** | 초기 화면에서 종료하기 버튼을 클릭하여 프로그램을 종료한다. |
| **Type** | System |
| **Cross References** | R1.3 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 프로그램이 종료된다. |

|  |  |
| --- | --- |
| **Name** | addCategory() |
| **Responsibilities** | Specification의 각 카테고리를 추가한다. |
| **Type** | System |
| **Cross References** | R2.1 |
| **Exceptions** | 같은 카테고리 이름이 존재하면 내부적으로 처리한다. |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 새로운 카테고리가 생성된다. |

|  |  |
| --- | --- |
| **Name** | delCategory() |
| **Responsibilities** | Specification의 각 카테고리를 삭제한다. |
| **Type** | System |
| **Cross References** | R2.1, R2.2, R2.3, R2.4 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 기존에 생성된 카테고리가 삭제된다. |

|  |  |
| --- | --- |
| **Name** | setCategory() |
| **Responsibilities** | Specification의 각 카테고리를 편집한다. |
| **Type** | System |
| **Cross References** | R2.1 |
| **Exceptions** | 같은 카테고리 이름이 존재하면 내부적으로 처리한다. |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 카테고리가 편집된다. |

|  |  |
| --- | --- |
| **Name** | addRepresentativeValue() |
| **Responsibilities** | Specification의 각 representative value를 생성한다 |
| **Type** | System |
| **Cross References** | R2.2 |
| **Exceptions** | 같은 representative value 이름이 존재하면 내부적으로 처리한다. |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | representative value가 생성된다. |

|  |  |
| --- | --- |
| **Name** | delRepresentativeValue() |
| **Responsibilities** | Specification의 각 representative value를 삭제한다 |
| **Type** | System |
| **Cross References** | R2.2, R2.3, R2.4 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | representative value가 삭제된다. |

|  |  |
| --- | --- |
| **Name** | setRepresentativeValue() |
| **Responsibilities** | Specification의 각 representative value를 편집한다 |
| **Type** | System |
| **Cross References** | R2.2 |
| **Exceptions** | 같은 representative value 이름이 존재하면 내부적으로 처리한다. |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | representative value가 편집된다. |

|  |  |
| --- | --- |
| **Name** | addProperty() |
| **Responsibilities** | representative value에 property constraint를 부여한다. |
| **Type** | System |
| **Cross References** | R2.3 |
| **Exceptions** | 같은 property 이름이 존재하면 내부적으로 처리한다. |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 해당 representative value에 property constraints가 부여된다. |

|  |  |
| --- | --- |
| **Name** | delProperty() |
| **Responsibilities** | representative value에 부여 된 property constraint를 삭제한다. |
| **Type** | System |
| **Cross References** | R2.3, R2.4 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 해당 representative value에 부여 된 property constraints가 제거된다. |

|  |  |
| --- | --- |
| **Name** | setProperty() |
| **Responsibilities** | representative value에 부여 된 property constraint를 편집한다. |
| **Type** | System |
| **Cross References** | R2.3 |
| **Exceptions** | 같은 property 이름이 존재하면 내부적으로 처리한다. |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 해당 representative value에 부여 된 property constraints가 수정된다. |

|  |  |
| --- | --- |
| **Name** | addIfProperty() |
| **Responsibilities** | representative value에 If-property constraint를 부여한다. |
| **Type** | System |
| **Cross References** | R2.4 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 해당 representative value에 If-property constraints가 부여된다. |

|  |  |
| --- | --- |
| **Name** | delIfProperty() |
| **Responsibilities** | representative value에 부여 된 If-property constraint를 삭제한다. |
| **Type** | System |
| **Cross References** | R2.4 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 해당 representative value에 부여 된 If-property constraints가 제거된다. |

|  |  |
| --- | --- |
| **Name** | setIfProperty() |
| **Responsibilities** | representative value에 부여 된 If-property constraint를 편집한다. |
| **Type** | System |
| **Cross References** | R2.4 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 해당 representative value에 부여 된 If-property constraints가 수정된다. |

|  |  |
| --- | --- |
| **Name** | setSingleError() |
| **Responsibilities** | representative value에 부여된 single 또는 error constrants를 편집한다. |
| **Type** | System |
| **Cross References** | R2.5 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 해당 representative value에 부여된 single 또는 error constrants가 수정된다. |

|  |  |
| --- | --- |
| **Name** | setPriorityRank() |
| **Responsibilities** | representative value에 부여된 priority를 편집한다. |
| **Type** | System |
| **Cross References** | R2.6 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 입력된 representative value들에 priority(중요도)가 부여된다. |

|  |  |
| --- | --- |
| **Name** | generateTestCases() |
| **Responsibilities** | Specification이 확인되면 test case generate가 시작된다. |
| **Type** | System |
| **Cross References** | R3 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 입력된 specification을 통해 test case를 만들고 그 총 개수가  화면을 통해 사용자에게 전달된다. |

|  |  |
| --- | --- |
| **Name** | export2excel() |
| **Responsibilities** | 생성된 Test cases를 엑셀 파일로 저장한다. |
| **Type** | System |
| **Cross References** | R4 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | Test case generate가 끝난 후 생성된 test cases들이 엑셀 파일로 저장 된다. 생성된 엑셀파일은 설정한 Priority순으로 정렬 된다. |

|  |  |
| --- | --- |
| **Name** | saveSpecification() |
| **Responsibilities** | 현재까지의 작업 상태를 저장한다. |
| **Type** | System |
| **Cross References** | R5 |
| **Exceptions** | N/A |
| **Output** | N/A |
| **Pre­conditions** | N/A |
| **Post­conditions** | 현재까지의 작업 상태가 specification 파일로 저장된다. 나중에 Load specification으로 불러올 수 있다. |

# Activity2038. Refine System Test Case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test**  **Number** | **Test**  **항목** | **Description** | **Use Case** | **System**  **Function** |
| 1-1 | Make new file  Test | 새로 만들기’ 버튼을 눌렀을 때, 생성될 파일의 경로를 입력하는 대화상자가 제대로 출력되는지 확인한다. | New specification file | R1.1 |
| 1-2 | Make new file  Test | 입력한 경로에 specification 파일이 제대로 생성되는지 확인한다. | New specification file | R1.1 |
| 1-3 | Make new file  Test | 기존에 있는 파일명을 입력했을 때 중복되는 이름에 대한 처리(덮어쓰기 혹은 생성금지)를 잘 하는지 확인한다. | New specification file | R1.1 |
| 1-4 | Make new file  Test | 입력 후 명세를 작성하는 메인 프레임이 제대로 뜨는지 확인한다 | New specification file | R1.1 |
| 2-1 | Load file Test | ‘불러오기’ 버튼을 눌렀을 때, 파일 탐색기가 제대로 출력되는지 확인한다. | Load specification file | R1.2 |
| 2-2 | Load file Test | 명세 파일 선택 시 제대로 명세 파일이 프로그램으로 로드 되는지 확인한다. | Load specification file | R1.2 |
| 2-3 | Load file Test | 명세 파일이 아닌 파일을 선택 시 경고 창과 함께 에러 핸들링을 했는지 확인한다. | Load specification file | R1.2 |
| 2-4 | Load file Test | 최근 사용했던 명세 파일 목록이 잘 유지되는지 확인한다. | Load specification file | R1.2 |
| 2-5 | Load file Test | 최근 파일 목록 중 하나를 선택하여 그 작업환경을 그대로 불러오는지 확인한다. | Load specification file | R1.2 |
| 3 | Shut down program Test | ‘종료’ 버튼을 눌렀을 때 정상적으로 종료되는지 확인한다. | Shut down program | R1.3 |
| 4-1 | Set category Test | 카테고리 추가를 눌렀을 때, 카테고리의 입력 및 생성이 원활한지 확인한다. | Set category | R2.1 |
| 4-1 | Set category Test | representative values, constraints를 다 설정해주고 삭제를 눌렀을 경우, 하위 항목(representative value, constraints)들까지 제대로 삭제되는지 확인한다. | Set category | R2.1 |
| 4-1 | Set category Test | Category가 하나도 존재하지 않을 때 다음 단계로 넘어가지 않게 제어하는지 확인한다. | Set category | R2.1 |
| 5-1 | Set representative value Test | 값 추가를 눌렀을 때, 각 카테고리 별로 representative values를 입력 받는 폼이 제대로 생성되는지 확인한다. | Set representative value | R2.2 |
| 5-2 | Set representative value Test | constraints를 설정해준 뒤 삭제를 눌렀을 경우, 하위 항목(constraints)이 제대로 삭제되는지 확인한다. | Set representative value | R2.2 |
| 5-3 | Set representative value Test | 어느 Category라도 representative value가 하나도 존재하지 않을 때 다음 단계로 넘어가지 않게 제어하는지 확인한다 | Set representative value | R2.2 |
| 6-1 | Set property Test | property 추가를 눌렀을 때, 각representative value 별로 property constraints를 입력 받는 폼이 제대로 생성되는지 확인한다. | Set property | R2.3 |
| 6-2 | Set property Test | 삭제를 눌렀을 때, 연관된 if constraints를 같이 삭제해 주는지 확인한다. | Set property | R2.3 |
| 7-1 | Set if-property Test | 설정 가능한 if 목록에 이전에 설정해 주었던 property들이 잘 나타나는지 확인한다. | Set if-property | R2.4 |
| 7-2 | Set if-property Test | if constraints의 삭제가 잘 이루어지는지 확인한다. | Set if-property | R2.4 |
| 8 | Set single and error Test | 각 representative value 별로 single constraints, error constraints가 잘 추가되고 삭제되는지 확인한다. | Set single and error | R2.5 |
| 9 | Set priority Test | 각 representative value 별 중요도를 제한된 범위 내에서 부여, 변경 및 삭제가 가능한지 확인한다. | Set priority | R2.6 |
| 11-4 | Move page Test | 단계에서 필요한 사항을 입력하지 않았을 때 넘어가지 못하게 제어가 잘 되는지 확인한다. | Move page | R2.8 |
| 12-1 | Generate test cases Test | Category Partitioning Test 알고리즘에 의해 정상적으로 테스트 케이스가 구해지는지 확인한다. | Generate test cases | R3 |
| 12-2 | Generate test cases Test | 테스트 케이스 수를 화면을 통해 사용자에게 잘 전달하는지 확인한다 | Generate test cases | R3 |
| 13-1 | Export test case to excel file Test | 생성된 테스트 케이스를 엑셀파일로 잘 저장해내는지 확인한다. | Export test case to excel file | R 4 |
| 13-2 | Export test case to excel file Test | 설정한 중요도에 따라 테스트 케이스가 정렬되는지 확인한다 | Export test case to excel file | R 4 |
| 14 | Save contemporary specification file Test | 저장 버튼을 누르거나, 최종적으로 종료될 때, 현재까지의 명세가 맨 처음 생성된 혹은 불러온 명세파일에 정상적으로 저장되는지 확인한다. | Save contemporary specification file | R5 |

# Activity2039. Traceability Analysis

