**Unit Testing Plan for Public Transportation System**

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

Project Team

**Team 3**

Date

**2014-11-21**

**Team Information**

**201311269 김제헌**

1. **박상희**
2. **박형민**

**201311287 엄현식**

**Table of Contents**

[1 Introduction 5](#_Toc76126)

[1.1 Objectives 5](#_Toc76127)

[1.2 Background 5](#_Toc76128)

[1.3 Scope 5](#_Toc76129)

[1.4 Project plan 5](#_Toc76130)

[1.5 Configuration management plan 5](#_Toc76131)

[1.6 References 5](#_Toc76132)

[2 Test items 6](#_Toc76133)

[2.1 Public Transportation System 6](#_Toc76134)

[2.2 Recharger System 7](#_Toc76135)

[2.3 Fee Calculation System 8](#_Toc76136)

[3 Features to be tested 8](#_Toc76137)

[3.1 Public Transportation System 8](#_Toc76138)

[3.2 Recharger System 9](#_Toc76139)

[3.3 Fee Calculation System 9](#_Toc76140)

[4 Features not to be tested 10](#_Toc76141)

[4.1 Public Transportation System 10](#_Toc76142)

[4.2 Recharger System 11](#_Toc76143)

[4.3 Fee Calculation System 11](#_Toc76144)

[5 Approach 11](#_Toc76145)

[6 Item pass/fail criteria 11](#_Toc76146)

[7 Unit test design specification 12](#_Toc76147)

[7.1 Test design specification identifier 12](#_Toc76148)

[7.1.1 Public Transportation System 12](#_Toc76149)

[7.1.2 Recharger System 12](#_Toc76150)

[7.1.3 Fee Calculation 12](#_Toc76151)

[7.2 Features to be tested 12](#_Toc76152)

[7.2.1 Process in SRA 12](#_Toc76153)

[7.2.1.1 Public Transportation System 12](#_Toc76154)

[7.2.1.2 Recharger System 12](#_Toc76155)

[7.2.1.3 Fee Calculation System 12](#_Toc76156)

[7.3 Approach refinements 12](#_Toc76157)

[7.3.1 Brute force Testing 12](#_Toc76158)

[7.4 Test identification 12](#_Toc76159)

[7.4.1 Public Transportation System 12](#_Toc76160)

[7.4.2 Recharger System 17](#_Toc76161)

[7.4.3 Fee Calculation System 18](#_Toc76162)

[7.5 Feature pass/fail criteria 21](#_Toc76163)

[8 Unit test case specification 22](#_Toc76164)

[8.1 Test case specification identifier 22](#_Toc76165)

[8.1.1 Public Transportation System 22](#_Toc76166)

[8.1.2 Recharger System 26](#_Toc76167)

[8.1.3 Fee Calculation System 28](#_Toc76168)

[8.2 Test items 31](#_Toc76169)

[8.2.1 Public Transportation System 31](#_Toc76170)

[8.2.2 Recharger System 31](#_Toc76171)

[8.2.3 Fee Calculation System 31](#_Toc76172)

[8.3 Input specifications 31](#_Toc76173)

[8.3.1 Public Transportation System 31](#_Toc76174)

[8.3.2 Recharger System 31](#_Toc76175)

[8.3.3 Fee Calculation System 31](#_Toc76176)

[8.4 Output specifications 32](#_Toc76177)

[8.4.1 Public Transportation System 32](#_Toc76178)

[8.4.2 Recharger System 32](#_Toc76179)

[8.4.3 Fee Calculation System 32](#_Toc76180)

[9 Testing tasks 32](#_Toc76181)

[10 Environmental needs 32](#_Toc76182)

[11 Unit Test deliverables 32](#_Toc76183)

[12 Schedules 32](#_Toc76184)

# 1 Introduction

## 1.1 Objectives

이 문서는 T3의 Public Transportation System, Recharger System, Fee Calculation System

의 unit test를 수행하기 위해 작성된 계획 문서이며, 본 system이 제대로 작동하는 지를 살펴보기 위해 필요한 요소들을 정리해 놓은 문서이다. Test를 수행하기 위 해 필요한 활동 및 자원을 정의하고, test approach 및 techniques를 정의한다. 또한 test 를 위한 환경적인 요구사항 및 test 도구들을 정의한다.

## 1.2 Background

Public Transportation System(이하 PTS)은 대중교통 카드 시스템으로, 사용자가 카드 태 그를 통해 요금 결제, 환승 및 카드 정보를 출력하는 시스템이다.

Recharger System(이하 GS)은 충전 시스템으로, 사용자가 카드를 태그하고 돈을 넣어 카드를 충전 하는 시스템이다.

Fee Calculation System(이하 FCS)은 정산 시스템으로, 3분마다 수익을 계산하고 수익을 출력하고 전송하는 시스템이다.

Unit test는 시스템을 구성하는 최소 단위 모듈들을 대상으로 하는 test이며, 시스템의 성 능을 좌우하는 요소들이 요구사항을 만족하는지를 확인할 수 있는 기본적인 Test

approach이다.

## 1.3 Scope

이 계획 문서는 PTS, GS, FCS의 unit test를 수행하기 위한 모든 것을 포함한다. PTS, GS, FCS의 unit test를 수행하기 위한 자원과 절차, test approach와 technique과 필요로 하는 환경 및 도구 등을 정의한다. PTS, GS, FCS의 unit test는 시스템을 구성하는 최소 단위의 모듈들을 대상으로 하며, 구현된 모듈이 요구사항을 만족하는지를 test 한다.

## 1.4 Project plan

## 1.5 Configuration management plan

## 1.6 References

DS-2014SE-PTS-SRS-1.0

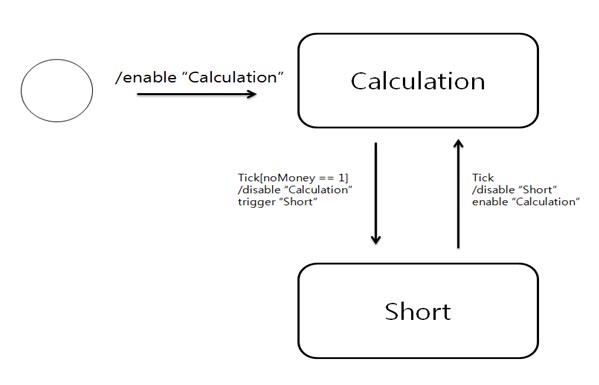
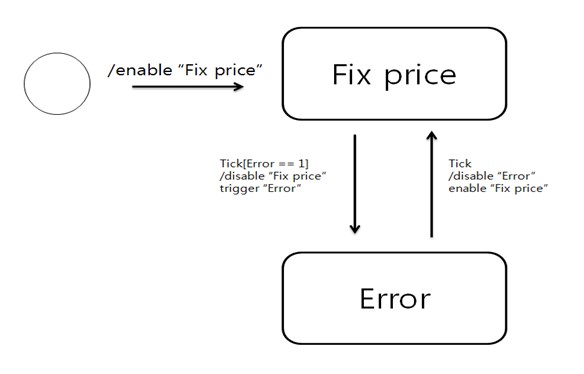
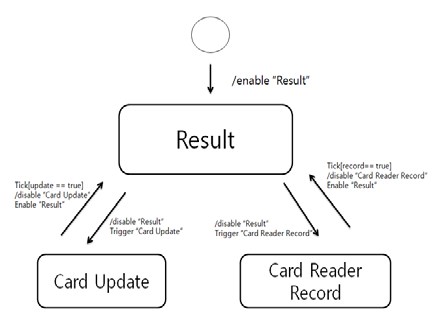
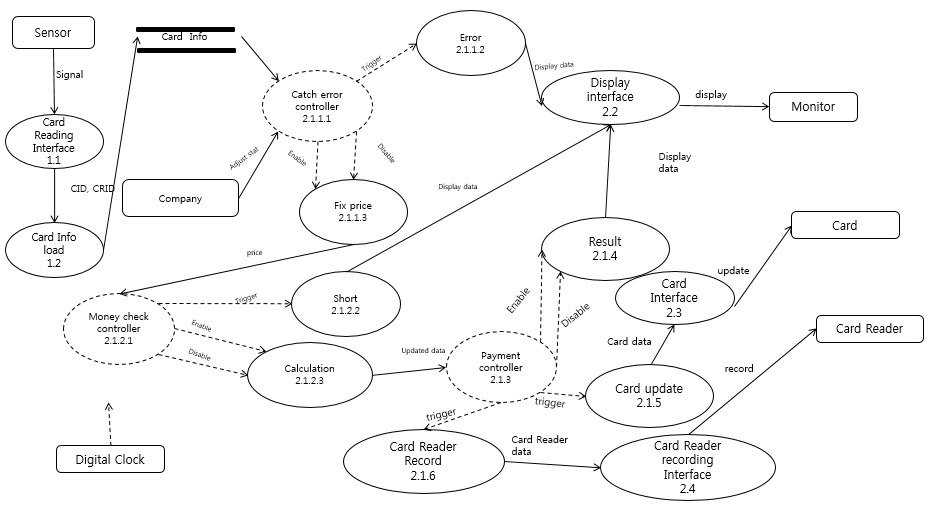
T3-2014-PTS-SRA-1.3

T3-2014-PTS-SDS-2.0

# 2 Test items

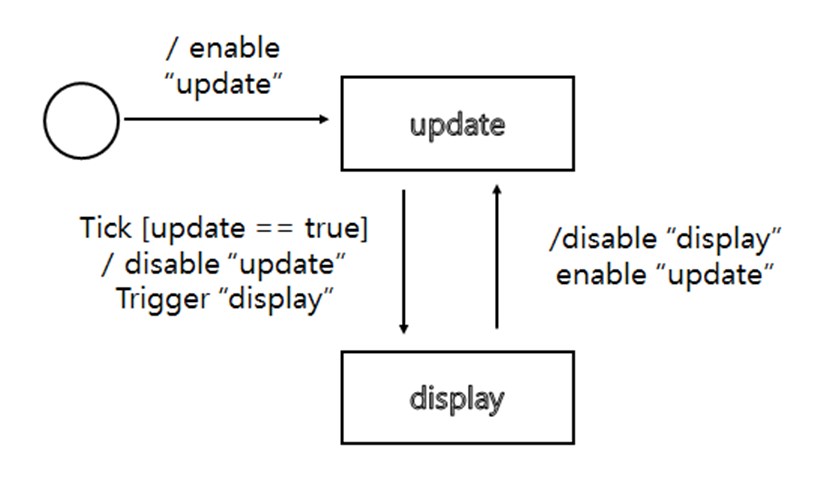
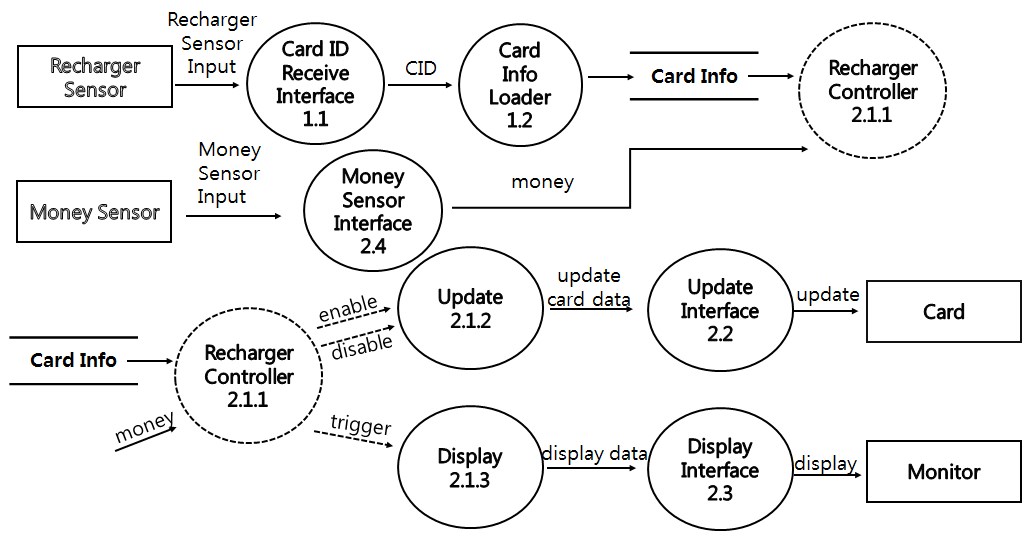
## 2.1 Public Transportation System

PTS를 구성하는 최소 단위의 모듈들이 unit test의 대상이 된다. 각 모듈의 요구사항을 만족 하는지를 test하며 test item은 다음 자료들로부터 작성되었다.



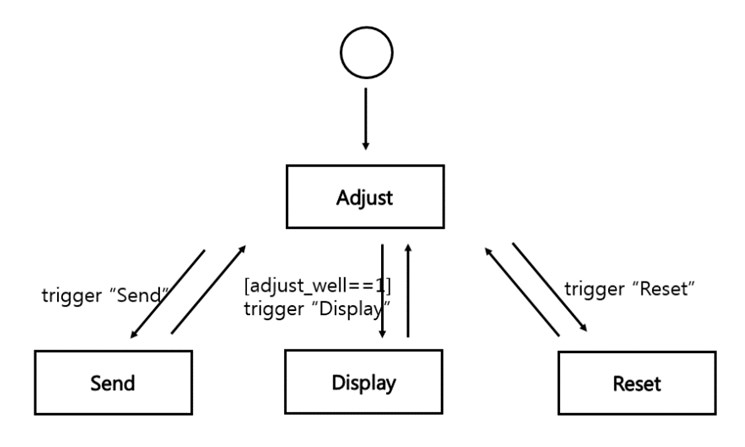
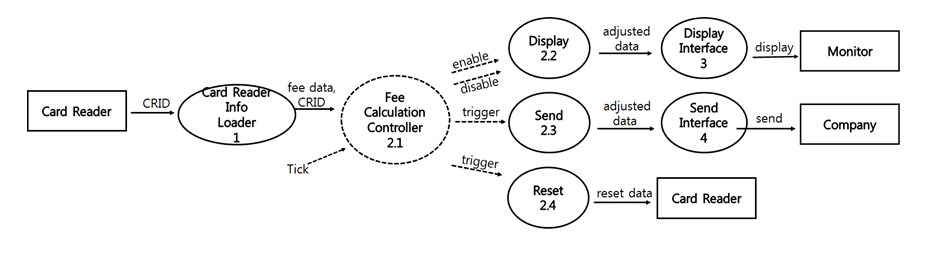
## 2.2 Recharger System

RS를 구성하는 최소 단위의 모듈들이 unit test의 대상이 된다. 각 모듈의 요구사항을 만족 하는지를 test하며 test item은 다음 자료들로부터 작성되었다.



## 2.3 Fee Calculation System

Fee Calculation System을 구성하는 최소 단위의 모듈들이 Unit Test의 대상이 된다. 각 모듈의 요구 사항을 만족하는지를 Test하며, Test Item은 다음 자료들로부터 작성 되었다



# 3 Features to be tested

1. Process in SRA : 각 프로세스가 가지고 있는 요구사항을 만족하는지를 Test한다.
2. Modules in SDS : 각 모듈이 가지고 있는 데이터 인터페이스를 Test한다.

## 3.1 Public Transportation System

##### <Table 1.1 테스트할 Process(DFD) 리스트>

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Description** |
| 1.2 | Card Info load | CID를 indexing하여 해당 Card의 잔액, 탑승 단말기, 승/하차, 태그 시간, 환승상태 정보를 불러온다. |
| 2.1.1.1 | Catch  Error  Controller | 정산 여부에 관한 정보를 받아온 뒤, 정산이 이루어지지 않았을 경우 경 고메시지를 출력하도록 한다.  정산이 이루어졌을 경우 Card Info를 받아와서 정상적인 카드 Tag가 이루 어졌는지 판별하고, 아닐 경우 경고메시지를 출력하도록 한다. 정상적일 |
|  |  | 경우 가격을 측정하도록 한다. |
| 2.1.1.2 | Error | 정상적이지 않은 Tag나 정산이 이루어지지 않았을 경우 경고 메시지를 보낸다. |
| 2.1.1.3 | Fix price | Card Info를 받아온 뒤, 해당 조건에 맞는 가격을 측정한다. |
| 2.1.2.1 | Money  Check  Controller | 측정된 가격을 받아온 뒤, Card Info의 잔액과 비교한 뒤 부족하면 Short, 충분하면 Calculation을 실행한다. |
| 2.1.2.2 | Short | 가격이 부족할 경우 경고메시지를 출력한다. |
| 2.1.2.3 | Calculation | 잔액이 충분할 경우 잔액-결제금액 을 한 뒤, 승/하차 상태 및 환승 상태, 역 단말기 정보, 잔액을 갱신한 뒤 Updated data로 보내준다. |
| 2.1.3 | Payment  Controller | Card 결제가 이루어 진 후, 처리된 정보와 현재 시각을 받아와서 결과값 을 출력하고, Card에 갱신, 역 단말기에 기록한다. |
| 2.1.4 | Result | Card 결제가 이루어지고 난 뒤의 결제 금액, 잔액, 현재 시각( Display data)을 보내준다. |
| 2.1.5 | Card update | Card 결제가 이루어지고 난 뒤 바뀐 Card Info를 갱신한다. |
| 2.1.6 | Card  Reader  Record | Card 결제가 이루어지고 난 뒤, 결제 금액을 Card Reader에 기록한다. |

## 3.2 Recharger System

**<Table 2.1 테스트할 Process(DFD) 리스트>**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Description** |
| 1.2 | Card Info  Loader | CID를 사용해서 Card Info를 불러와 Recharger Control 에 전달한다. |
| 2.1.1 | Recharger  Controller | 입력받은 Card Info, money Data를 종합하여 충전 계산을 한 뒤, 적절한 update와 display를 실행 해준다. |
| 2.1.2 | Update | 충전된 정보로 교통카드를 갱신한다. |
| 2.1.3 | Display | 교통카드에 충전된 정보를 Monitor에 보여준다. |

## 3.3 Fee Calculation System

<**Table 3.1 테스트할 Process(DFD) 리스트**>

|  |  |  |
| --- | --- | --- |
| ID | Name | Description |
| 1 | Card Reader Info Loader | CRID(단말기 고유 ID)를 받아서 해당하는 단말기파일을 열 |
|  |  | 어서 데이터들을 불러와서 데이터 형식에 맞게 각각 저장한 다. |
| 2.1 | Fee Calculation Controller | CRID를 통해서 얻은 데이터와 Tick을 받아서 정산을 한 후 상태를 통해서 적절한 프로세스에 명령을 전달한다. |
| 2.2 | Display | Fee Calculation Controller가 정산한 결과를 출력할 때 호출 하는 프로세스. 명령을 받으면 adjust\_well==0인지 비교(정 산이 잘 되었는지)를 하여 정산이 잘 되었으면 adjusted data (fee\_bus, fee\_metro, time\_now)를 출력한다. |
| 2.3 | Send | Fee Calculation Controller가 정산한 결과를 지하철회사와 버 스회사에 보낼 때 호출하는 프로세스. Trigger 명령을 받으면 adjust\_well==0인지 비교를 하여 정산이 잘 되었으면 버스 회사와 지하철 회사에 정산이 잘되었다는 값(c\_well)과 정산 결과값(fee\_bus or fee\_metro)을 보낸다. |
| 2.4 | Reset | Fee Calculation Controller가 정산한 결과를 초기화 시킬 때 호출하는 프로세스. Trigger 명령을 받으면 adjust\_well==0인 지 비교를 하여 정산이 잘 되었으면 모든 단말기 파일을 초 기화 시킨다. |

# 4 Features not to be tested

Process in SRA : 외부 장치 드라이버, 단순 데이터 전달 프로세스 등은 Test에서 제외한다.

## 4.1 Public Transportation System

**<Table 1.2 테스트하지 않을 Process(DFD) 리스트>**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Description** |
| 1.1 | Card  Reading  Interface | 카드를 Tag했을 때 전달되는 신호를, 컴퓨터가 해석할 수 있는 값을 바꾸어 보내준다. |
| 2.2 | Display interface | 결정된 금액 혹은 경고 메시지, 현재 시각에 관한 정보를 Display data 를 통해 받아오고, 정리된 정보를 보내준다. |
| 2.3 | Card interface | 결제 후, 갱신해야 할 Card data를 받아온 뒤, Card에 넘겨준다. |
| 2.4 | Card  Reader Recording interface | 결제 후, 결제 금액을 받아온 뒤, 각 Card Reader에 기록하기 위한 정 보를 보내준다. |

## 4.2 Recharger System

##### <Table 2.2 테스트하지 않을 Process(DFD) 리스트>

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** |  | **Description** |
| 1.1 | Card  Receive  Interface | ID | Recharger Sensor로부터 받은 아날로그 신호를 디지털 신호로 변환한 다 |
| 2.2 | Update  Interface |  | update card data를 받아서 Card 정보를 Update 시키는 정보를 보내 준다 |
| 2.3 | Display interface |  | display data를 받아서 Monitor에 출력할 display정보를 보내준다. |
| 2.4 | Money  Sensor  Interface |  | Money Sensor로부터 받은 아날로그 신호를 디지털 신호로 변환한다. |

## 4.3 Fee Calculation System

##### <Table 3.2 테스트하지 않을 Process(DFD) 리스트>

|  |  |  |
| --- | --- | --- |
| ID | Name | Description |
| 3 | Display Interface | adjusted data(fee\_bus, fee\_metro, time\_now)를 받아서 정산 결과를 출력해주는 Display 장치에 보내준다. |
| 4 | Send Interface | adjusted data(fee\_bus, fee\_metro, c\_well)를 받아서 정산 결 과를 버스 회사와 지하철 회사에 보내주는 Send 장치에 보 내준다. |

# 5 Approach

Public Transportation System, Recharger System, Fee Calculation System의 Program source code 및 Unit Test를 위한 Test code는 Visual studio 2010 환경에서 이루어지며, Program source code/ Test code의 변경 및 수정 사항은 지속적으로 통합되고 Test된다.

1)Brute force Testing : 각 모듈의 요구사항을 만족하는지를 확인할 수 있는 Test Case를 작성 한다. 그 이외의 예외사항에 대해서는 Test하지 않는다.

# 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

### 7.1.1 Public Transportation System

PTS.UTC.0000.000

### 7.1.2 Recharger System

RS.UTC.000.000

### 7.1.3 Fee Calculation

FCS.UTC.000.000

## 7.2 Features to be tested

### 7.2.1 Process in SRA

#### 7.2.1.1 Public Transportation System

**<Table 1.1 테스트할 Process(DFD) 리스트> 참조**

#### 7.2.1.2 Recharger System

**<Table 2.1 테스트할 Process(DFD) 리스트> 참조**

#### 7.2.1.3 Fee Calculation System

**<Table 3.1 테스트할 Process(DFD) 리스트> 참조**

## 7.3 Approach refinements

### 7.3.1 Brute force Testing

PTS, RS, FCS의 각 모듈이 요구사항을 만족하는지를 확인하기 위하여, 요구사항에 정의된 내용에 기반하여 Test Case를 작성한다. 단, 그 이외의 상황에 대해서는 Test Case를 작성하지 않는다.

## 7.4 Test identification

### 7.4.1 Public Transportation System

<**Table 1.3 Test Design Identification**>

|  |  |  |
| --- | --- | --- |
| **Identifier** | **Feature** | **Valid/Invalid Value** |
| PTS\_UTC\_1200\_000 | 1.2 Card Info Loader | File!=NULL 상태에서 유효한 card\_info의 입력이 들어온다. |
| PTS\_UTC\_1200\_001 | 1.2 Card Info Loader | File!=NULL 유효하지 않은 card\_info의 입력이 들 어온다. |
| PTS\_UTC\_1200\_002 | 1.2 Card Info Loader | File==NULL인 입력이 들어온다. |
| PTS\_UTC\_2111\_000 | 2.1.1.1 Catch Error  Controller | card\_info의 값과 정수인 price의 값과 stat==Normal인 입력이 들어온다 |
| PTS\_UTC\_2111\_001 | 2.1.1.1 Catch Error  Controller | card\_info의 값과 정수인 price의 값과 stat!=Normal 인 입력이 들어온다 |
| PTS\_UTC\_2112\_000 | 2.1.1.2 Error | interval\_sec <=0 인 입력이 들어온다. |
| PTS\_UTC\_2112\_001 | 2.1.1.2 Error | state==1,interval\_sec<15,getout==0이고 CRID의 값이 card\_info의 CRID의 값과 같은 입력이 들어 온다. |
| PTS\_UTC\_2112\_002 | 2.1.1.2 Error | state==1,interval\_sec>=15,getout==0이고 CRID의 값이 card\_info의 CRID의 값과 같은 입력이 들어 온다. |
| PTS\_UTC\_2112\_003 | 2.1.1.2 Error | state==1,interval\_sec<15,getout!=0이고 CRID의 값이 card\_info의 CRID의 값과 같은 입력이 들어 온다. |
| PTS\_UTC\_2112\_004 | 2.1.1.2 Error | state==1,interval\_sec<15,getout==0이고 CRID의 값이 card\_info의 CRID의 값과 다른 입력이 들어 온다. |
| PTS\_UTC\_2112\_005 | 2.1.1.2 Error | state==1,interval\_sec<15,getout!=0이고 CRID의 값이 card\_info의 CRID의 값과 다른 입력이 들어 온다. |
| PTS\_UTC\_2112\_006 | 2.1.1.2 Error | state==1,interval\_sec>=15,getout!=0이고 CRID의 값이 card\_info의 CRID의 값과 같은 입력이 들어  온다. |
| PTS\_UTC\_2112\_007 | 2.1.1.2 Error | state==1,interval\_sec>=15,getout==0이고 CRID의 값이 card\_info의 CRID의 값과 다른 입력이 들어 온다. |
| PTS\_UTC\_2112\_008 | 2.1.1.2 Error | state==1,interval\_sec>=15,getout!=0이고 CRID의 값이 card\_info의 CRID의 값과 다른 입력이 들어  온다. |
| PTS\_UTC\_2112\_009 | 2.1.1.2 Error | state==0,interval\_sec<15,getout==0이고 CRID의 값이 card\_info의 CRID의 값과 같은 입력이 들어 |

|  |  |  |
| --- | --- | --- |
|  |  | 온다. |
| PTS\_UTC\_2112\_010 | 2.1.1.2 Error | state==0,interval\_sec>=15,getout==0이고 CRID의 값이 card\_info의 CRID의 값과 같은 입력이 들어 온다. |
| PTS\_UTC\_2112\_011 | 2.1.1.2 Error | state==0,interval\_sec<15,getout!=0이고 CRID의 값이 card\_info의 CRID의 값과 같은 입력이 들어 온다. |
| PTS\_UTC\_2112\_012 | 2.1.1.2 Error | state==0,interval\_sec<15,getout==0이고 CRID의 값이 card\_info의 CRID의 값과 다른 입력이 들어 온다. |
| PTS\_UTC\_2112\_013 | 2.1.1.2 Error | state==0,interval\_sec<15,getout!=0이고 CRID의 값이 card\_info의 CRID의 값과 다른 입력이 들어 온다. |
| PTS\_UTC\_2112\_014 | 2.1.1.2 Error | state==0,interval\_sec>=15,getout!=0이고 CRID의 값이 card\_info의 CRID의 값과 같은 입력이 들어 온다. |
| PTS\_UTC\_2112\_015 | 2.1.1.2 Error | state==0,interval\_sec>=15,getout==0이고 CRID의 값이 card\_info의 CRID의 값과 다른 입력이 들어 온다. |
| PTS\_UTC\_2112\_016 | 2.1.1.2 Error | state==0,interval\_sec>=15,getout!=0이고 CRID의 값이 card\_info의 CRID의 값과 다른 입력이 들어 온다. |
| PTS\_UTC\_2113\_000 | 2.1.1.3 FixPrice | interval\_sec <= 0인 입력을 들어온다. |
| PTS\_UTC\_2113\_001 | 2.1.1.3 FixPrice | CRID < 10인 입력이 들어온다. |
| PTS\_UTC\_2113\_002 | 2.1.1.3 FixPrice | CRID >= 10인 입력이 들어온다. |
| PTS\_UTC\_2113\_003 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 1 && transfer == 1 && tp == 1이다. |
| PTS\_UTC\_2113\_004 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 1 && transfer == 1 && tp == 0이다. |
| PTS\_UTC\_2113\_005 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 1 && transfer == 1 && tp != 1 && tp != 0이다. |
| PTS\_UTC\_2113\_006 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 1 && transfer == 0 && tp == 1이다. |
| PTS\_UTC\_2113\_007 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 1 && transfer == 0 && tp == 0이다. |
| PTS\_UTC\_2113\_008 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 1 && transfer == 0 && tp != 1 && tp != 0이다. |
| PTS\_UTC\_2113\_009 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 1 && |

|  |  |  |
| --- | --- | --- |
|  |  | transfer != 0 && transfer != 1이다. |
| PTS\_UTC\_2113\_010 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 0 && transfer == 0 && interval\_sec <= 15 && tp == card\_info->tp이다. |
| PTS\_UTC\_2113\_011 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 0 && transfer == 0 && interval\_sec <= 15 && tp == 1 && card\_info->tp == 0이다. |
| PTS\_UTC\_2113\_012 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 0 && transfer == 0 && interval\_sec <= 15 && tp == 0 && card\_info->tp == 1이다. |
| PTS\_UTC\_2113\_013 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 0 && transfer == 0 && interval\_sec <= 15 && tp != 0  && tp != 1 && card\_info->tp != 0 && card\_info>tp != 1이다. |
| PTS\_UTC\_2113\_014 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 0 &&  transfer == 0 && interval\_sec > 15 |
| PTS\_UTC\_2113\_015 | 2.1.1.3 FixPrice | (CRID%10) == 1 상태에서 state == 0 && transfer != 0 |
| PTS\_UTC\_2113\_016 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 &&  transfer == 1 && tp == 1 이면서 interval\_station 가 1 또는 4이다. |
| PTS\_UTC\_2113\_017 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 &&  transfer == 1 && tp == 1 이면서 interval\_station 가 2 또는 3이다. |
| PTS\_UTC\_2113\_018 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 &&  transfer == 1 && tp == 1 이면서 interval\_station 가 1~4가 아니다. |
| PTS\_UTC\_2113\_019 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 && transfer == 1 && tp == 0 이면서  (interval\_sec/30) > 5이다. |
| PTS\_UTC\_2113\_020 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 && transfer == 1 && tp == 0 이면서  (interval\_sec/30) <= 5이다. |
| PTS\_UTC\_2113\_021 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 && transfer == 1 && tp != 0 && tp != 1이다. |
| PTS\_UTC\_2113\_022 | 2.1.1.3 FixPrice | PTS\_UTC\_2113\_021 (CRID%10) == 0 상태에서 state == 1 && transfer == 0 && tp == 1 이면서 interval\_station가 1 또는 4이다. |

|  |  |  |
| --- | --- | --- |
| PTS\_UTC\_2113\_023 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 &&  transfer == 0 && tp == 1 이면서 interval\_station 가 2 또는 3이다. |
| PTS\_UTC\_2113\_024 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 &&  transfer == 0 && tp == 1 이면서 interval\_station 가 1~4가 아니다. |
| PTS\_UTC\_2113\_025 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 && transfer == 0 && tp == 0 이다. |
| PTS\_UTC\_2113\_026 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state == 1 && transfer == 0 && tp != 1 && tp != 0 이다. |
| PTS\_UTC\_2113\_027 | 2.1.1.3 FixPrice | (CRID%10) == 0 상태에서 state != 1 이다. |
| PTS\_UTC\_2113\_028 | 2.1.1.3 FixPrice | (CRID%10) != 0 && (CRID%10) != 1 이다. |
| PTS\_UTC\_2121\_000 | 2.1.2.1 Money Check  Controller | stat == Normal이다. |
| PTS\_UTC\_2121\_001 | 2.1.2.1 Money Check  Controller | Short호출 후 stat == Normal이다. |
| PTS\_UTC\_2122\_000 | 2.1.2.2 Short | (cash-price) < 0 이다. |
| PTS\_UTC\_2122\_00  0 | 2.1.2.3 Calculation | CRID < 10 값이 들어온다. |
| PTS\_UTC\_2123\_001 | 2.1.2.3 Calculation | CRID >= 10 값이 들어온다. |
| PTS\_UTC\_2123\_002 | 2.1.2.3 Calculation | 카드에 저장된 transfer값이 0이다. |
| PTS\_UTC\_2123\_003 | 2.1.2.3 Calculation | (CRID%10)==0이다. |
| PTS\_UTC\_2123\_004 | 2.1.2.3 Calculation | (CRID%10)!=0이다. |
| PTS\_UTC\_2130\_000 | 2.1.3 Payment  Controller | stat == Normal 값이 들어온다. |
| PTS\_UTC\_2140\_000 | 2.1.4 Result | stat == Normal && transfer == 1인 값이 들어온 다. |
| PTS\_UTC\_2140\_001 | 2.1.4 Result | Normal && transfer != 1인 값이 들어온다. |
| PTS\_UTC\_2140\_002 | 2.1.4 Result | stat == HoplnProcessing 값이 들어온다. |
| PTS\_UTC\_2140\_003 | 2.1.4 Result | stat == GetoffProcessing 값이 들어온다. |
| PTS\_UTC\_2140\_004 | 2.1.4 Result | stat == EShort 값이 들어온다. |
| PTS\_UTC\_2140\_005 | 2.1.4 Result | stat == NotAddjust 값이 들어온다. |
| PTS\_UTC\_2140\_006 | 2.1.4 Result | stat == InvalidInput 값이 들어온다. |
| PTS\_UTC\_2150\_000 | 2.1.5 Card update | file != NULL && newFile != NULL && feof(file) == false 상태에서 CID == fileCID 값이 들어온다. |
| PTS\_UTC\_2150\_001 | 2.1.5 Card update | file != NULL && newFile != NULL && feof(file) == false 상태에서 CID != fileCID 값이 들어온다. |
| PTS\_UTC\_2160\_000 | 2.1.6 Card Reader | file != NULL 이다. |
|  | Record |  |

### 7.4.2 Recharger System

**<Table 2.3 Test Design Identification>**

|  |  |  |
| --- | --- | --- |
| **Identifier** | **Feature** | **Valid/Invalid Value** |
| PTS\_UTC\_120\_000 | 1.2 Card Info Loader | Card.txt파일을 열어 입력받은 CID값이 있는 줄 을 찾고, 찾으면 카드의 정보를 덮어씌운다. |
| PTS\_UTC\_120\_001 | 1.2 Card Info Loader | Card.txt파일을 열어 입력받은 CID값이 있는 줄 을 찾고, 찾지 못하면 카드의 정보를 초기화해 준다. |
| PTS\_UTC\_120\_002 | 1.2 Card Info Loader | Card.txt파일이 지정한 상대경로에 존재하지 않 는다면, 파일열기 실패를 출력하면서 프로그램 이 종료된다. |
| PTS\_UTC\_211\_000 | 2.1.1 Recharger  Controller | 적합한 CID값을 입력받아서, card\_info값을 갱신 한 후에는, money값을 입력 받고, 잔액을 충전 한 후에, 충전된 정보로 교통카드를 갱신하고, 충전시각과 함께 교통카드에 충전된 정보를  Monitor에 보여준다. |
| PTS\_UTC\_211\_001 | 2.1.1 Recharger  Controller | 적합한 CID값을 입력받지 못해서, card\_info값을 0으로 모두 초기화한 후일지라도, money값을 입력 받고, 잔액을 충전한다. 충전된 정보로 교 통카드를 갱신하려 하지만, Card.txt파일에서 일 치하는 CID정보가 없으므로 실질적으로 갱신은 이루어지지 않는다. 이후, 충전시각과 함께 교 통카드에 충전된 정보를 Monitor에 보여준다. |
| PTS\_UTC\_212\_000 | 2.1.2. Update | 입력받은 충전할 금액과 카드의 잔액을 더한 카드의 정보를 입력 받아서, Card.txt파일과 newCard.txt파일을 열은 후, Card.txt파일에서,  입력받은 CID값이 있는 줄이면 충전한 카드정 보를 newCard.txt에 기록하고, 다른 CID값이 있 는 줄이면 그대로 newCard.txt에 기록한다.  이후, Card.txt 파일은 제거하고, newCard.txt의 이름을 Card.txt로 변경한다. |
| PTS\_UTC\_212\_001 | 2.1.2. Update | 적합한 CID값을 입력받지 못해서, card\_info값을 0으로 모두 초기화한 카드의 정보를 입력받을 경우, Card.txt파일에서, CID값이 0인 줄이 없으 |
|  |  | 므로, Card.txt파일의 값이 그대로 newCard.txt에  기록되면서 실질적으로 바뀌는 부분은 없다. 이 후, Card.txt 파일은 제거하고, newCard.txt의 이 름을 Card.txt로 변경한다. |
| PTS\_UTC\_212\_002 | 2.1.2. Update | Card.txt파일이 지정한 상대경로에 존재하지 않 는다면, 파일열기 실패를 출력하면서 프로그램 이 종료된다. |
| PTS\_UTC\_213\_000 | 2.1.3. Display | 적합한 CID값을 입력받아서, card\_info값을 갱신 한 후에는, 충전된 카드의 정보와 충전한 금액 을 입력 받았을 때, 충전한 시각을 구한 후, 충 전 한 시각, 충전 전의 금액, 충전 한 금액, 충 전 후의 금액을 보여준다. |
| PTS\_UTC\_213\_001 | 2.1.3. Display | 적합한 CID값을 입력받지 못해서, card\_info값을  0으로 모두 초기화한 카드가 충전된 정보와 충 전한 금액을 입력 받았을 때, 충전한 시각을 구 한 후, 충전 한 시각, 충전 전의 금액(=0원), 충 전 한 금액, 충전 후의 금액(=충전 한 금액)을 보여준다. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identifier | Feature | Valid/Invalid value | | |  |
| FCS.UTC.010.000 | Card Reader Info Loader | Tick()==0일 때, 각 단말기 파일 != NULL이면 모든 단말기 파일에 대하여 CardReader\_info!=NULL 입력이 들어온다. | | |
| FCS.UTC.010.001 | Card Reader Info Loader | Tick()==0일 때, 각 단말기 파일 == NULL이면 모든 단말기 파일에 대하여 CardReader\_info!=NULL 입력이 들어온다. | | |
| FCS.UTC.010.002 | Card Reader Info Loader | Tick()==1일 때, 각 단말기 파일 != NULL이면 모든 단말기 파일에 대하여 CardReader\_info!=NULL 입력이 들어온다. | | |
| FCS.UTC.010.003 | Card Reader Info Loader | Tick()==1일 때, 각 단말기 파일 == NULL이면 모든 단말기 파일에 대하여 CardReader\_info==NULL 입력이 들어온다. | | |
| FCS.UTC.010.004 | Card Reader Info Loader | CardReader\_info[i].CID==1000일 CID\_1000.txt에 저장한다. | 때, | CardReader\_info[i]를 |
| FCS.UTC.010.005 | Card Reader Info Loader | CardReader\_info[i].CID==1001일 CID\_1001.txt에 저장한다. | 때, | CardReader\_info[i]를 |  |
| FCS.UTC.010.006 | Card Reader Info Loader | CardReader\_info[i].CID==1002일 CID\_1002.txt에 저장한다. | 때, | CardReader\_info[i]를 |  |

### 7.4.3 Fee Calculation System

<**Table 3.3 Test Design Identification**>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FCS.UTC.010.007 | 1 Card Reader Info Loader | CardReader\_info[i].CID==1003일 CID\_1003.txt에 저장한다. | 때, | CardReader\_info[i]를 |
| FCS.UTC.010.008 | 1 Card Reader Info Loader | CardReader\_info[i].CID==1004일 CID\_1004.txt에 저장한다. | 때, | CardReader\_info[i]를 |
| FCS.UTC.010.009 | 1 Card Reader Info Loader | CardReader\_info[i].CID==1005일 CID\_1005.txt에 저장한다. | 때, | CardReader\_info[i]를 |
| FCS.UTC.010.010 | 1 Card Reader Info Loader | CardReader\_info[i].CID==1006일 CID\_1006.txt에 저장한다. | 때, | CardReader\_info[i]를 |
| FCS.UTC.010.011 | 1 Card Reader Info Loader | CardReader\_info[i].CID==NULL일 CID\_\*.txt에 저장한다. | 때, | CardReader\_info[i]를 |
| FCS.UTC.021.000 | 2.1 Fee Calculation Controller | AdjustStart상태 일 때, c\_well==1을 입력한다. | | |
| FCS.UTC.021.001 | 2.1 Fee Calculation Controller | AdjustStart상태 일 때, c\_well==0을 입력한다. | | |
| FCS.UTC.021.002 | 2.1 Fee Calculation Controller | GuestOut상태 일 때, state==1이면 getout==1을 입력한다. | | |
| FCS.UTC.021.003 | 2.1 Fee Calculation Controller | GuestOut상태 일 때, state==1이면 getout==0을 입력한다. | | |
| FCS.UTC.021.004 | 2.1 Fee Calculation Controller | GuestOut상태 일 때, state==0이면 getout==1을 입력한다. | | |
| FCS.UTC.021.005 | 2.1 Fee Calculation Controller | CID\_Sort상태 일 때, CID\_\*.txt != NULL이면 CID\_\*.txt 데이 터를 CardReader\_info 배열에 저장한다. | | |
| FCS.UTC.021.006 | 2.1 Fee Calculation Controller | CID\_Sort상태 일 때, CID\_\*.txt == NULL이면 CID\_\*.txt 데이 터를 CardReader\_info 배열에 저장한다. | | |
| FCS.UTC.021.007 | 2.1 Fee Calculation Controller | CID\_Sort상태 일 때, CardReader\_info배열이 !=NULL이면 tagTime이 작은 순으로 real\_CardReader\_info 배열에 저 장한다. | | |
| FCS.UTC.021.008 | 2.1 Fee Calculation Controller | CID\_Sort상태 일 때, CardReader\_info배열이 ==NULL이면 tagTime이 작은 순으로 real\_CardReader\_info 배열에 저 장한다. | | |
| FCS.UTC.021.009 | 2.1 Fee Calculation Controller | UnAdjust상태 일 때, real\_CarfReader\_info!=NULL, line!=NULL, &bus\_fee!=NULL, &metro\_fee!=NULL이면 Adjust를 실행한다. | | |
| FCS.UTC.021.010 | 2.1 Fee Calculation Controller | UnAdjust상태 일 때, real\_CarfReader\_info==NULL || line==NULL || &bus\_fee==NULL || &metro\_fee==NULL이 면 Adjust를 실행한다. | | |
| FCS.UTC.021.011 | 2.1 Fee Calculation Controller | Adjust상태 일 때, i>0, real\_CardReader\_info[i].price>1050면 real\_Card\_info[i].price += real\_Card\_info[i].price-1050, real\_Card\_info[i].price-=1050을 실행한다. | | |
| FCS.UTC.021.012 | 2.1 Fee Calculation Controller | Adjust상태 일 때, i>0, real\_CardReader\_info[i].price>1050면 real\_Card\_info[i].price += real\_Card\_info[i].price-1050, real\_Card\_info[i].price-=1050을 실행하지 않는다. | | |
| FCS.UTC.021.013 | 2.1 Fee Calculation Controller | Adjust상태 일 때, (배열의 처음)|| (배열의 마지막) | | |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | ||(state==1, transfer==0)이면 다음 state==1, transfer==0인 전까지를 하거나 state==1, transfer==0인 곳이 없으면 배  열의 처음이나 마지막이면 처음이나 마지막을 정산 계산을 할 구간을 정해서 index에 배열 인덱스를 넣는다. | |
| FCS.UTC.021.014 | 2.1 Fee Calculation Controller | Adjust상태 일 때, (배열의 처음)|| (배열의 마지막)  ||(state==1, transfer==0)이면 다음 state!=1, transfer!=0인 전까지 정산 계산을 할 구간을 정해서 index에 배열 인덱 스를 넣는다. | |
| FCS.UTC.021.015 | 2.1 Fee Calculation Controller | Adjust상태 일 때, index!=NULL이면 real\_fee+= real\_CardReader\_info[i].price를 수행한다. | |
| FCS.UTC.021.016 | 2.1 Fee Calculation Controller | Adjust상태 일 때, index==NULL이면 real\_fee+= real\_CardReader\_info[i].price를 수행한다. | |
| FCS.UTC.021.017 | 2.1 Fee Calculation Controller | Adjust상태 일 때, real\_CardReader\_info[i-1].tp != real\_CardReader\_info[i].tp이면 total\_fee+=real\_fee를 수행한 다. | |
| FCS.UTC.021.018 | 2.1 Fee Calculation Controller | Adjust상태 일 때, real\_CardReader\_info[i-1].tp == real\_CardReader\_info[i].tp이면 total\_fee+=real\_fee를 수행한 다. | |
| FCS.UTC.021.019 | 2.1 Fee Calculation Controller | Adjust상태 일 때, index!=NULL이면 temp\_fee+= real\_CardReader\_info[i].price를 수행한다. | |
| FCS.UTC.021.020 | 2.1 Fee Calculation Controller | Adjust상태 일 때, index==NULL이면 temp\_fee+= real\_CardReader\_info[i].price를 수행한다. | |
| FCS.UTC.021.021 | 2.1 Fee Calculation Controller | Adjust상태 일 때, ((real\_CardReader\_info[j-1].tp != real\_CardReader\_info[j].tp) || i==j), real\_CardReader\_info[j].tp==0이면  \*bus\_fee+=temp\_fee/total\_fee\*real\_fee를 수행한다. | |
| FCS.UTC.021.022 | 2.1 Fee Calculation Controller | Adjust상태 일 때, ((real\_CardReader\_info[j-1].tp == real\_CardReader\_info[j].tp) || i!=j), real\_CardReader\_info[j].tp==0이면  \*bus\_fee+=temp\_fee/total\_fee\*real\_fee를 수행한다. | |
| FCS.UTC.021.023 | 2.1 Fee Calculation Controller | Adjust상태 일 때, ((real\_CardReader\_info[j-1].tp != real\_CardReader\_info[j].tp) || i==j), real\_CardReader\_info[j].tp==1이면  \*metro\_fee+=temp\_fee/total\_fee\*real\_fee를 수행한다. | |
| FCS.UTC.021.024 | 2.1 Fee Calculation Controller | Adjust상태 일 때, ((real\_CardReader\_info[j-1].tp == real\_CardReader\_info[j].tp) || i!=j), real\_CardReader\_info[j].tp==1이면  \*metro\_fee+=temp\_fee/total\_fee\*real\_fee를 수행한다. | |
| FCS.UTC.021.025 | 2.1 Fee Calculation Controller | Index 구간이 끝나면, index=i-1, total\_feee=0, real\_fee=0, temp\_fee=0를 수행한다. | |
| FCS.UTC.021.026 | 2.1 Fee Calculation Controller | Index 구간이 끝나지 않아도, index=i-1, total\_feee=0, real\_fee=0, temp\_fee=0를 수행한다. | |
| FCS.UTC.021.027 | 2.1 Fee Calculation Controller | Adjus상태가 끝나고, bus\_fee!=NULL, metro\_fee!=NULL이면 c\_well=0을 수행한다. | |
| FCS.UTC.021.028 | 2.1 Fee Calculation Controller | Adjus상태가 끝나고, bus\_fee==NULL, metro\_fee==NULL이 면 c\_well=0을 수행한다. | |
| FCS.UTC.021.029 | 2.1 Fee Calculation Controller | c\_well==0면 enable Display. | |
| FCS.UTC.021.030 | 2.1 Fee Calculation Controller | c\_well!=0면 enable Display. | |
| FCS.UTC.021.031 | 2.1 Fee Calculation Controller | c\_well==0이면 Trigger Send | |
| FCS.UTC.021.032 | 2.1 Fee Calculation Controller | c\_well!=0이면 Trigger Send | |
| FCS.UTC.021.033 | 2.1 Fee Calculation Controller | c\_well==0이면 Trigger Reset | |
| FCS.UTC.021.034 | 2.1 Fee Calculation Controller | c\_well!=0이면 Trigger Reset |  |
| FCS.UTC.022.000 | 2.2 Display | Enable이 들어오고 c\_well==0이면 bus\_fee, t\_now를 print한다 | metro\_fee, |
| FCS.UTC.022.001 | 2.2 Display | Enable이 들어오고 c\_well!=0이면 bus\_fee, t\_now를 print한다 | metro\_fee, |
| FCS.UTC.022.002 | 2.2 Display | Enable이 들어오고 c\_well!=0이면 bus\_fee, t\_now를 print안한다 | metro\_fee, |
| FCS.UTC.023.000 | 2.3 Send | Trigger가 들어오고 c\_well==0이면 c\_well, metro\_fee를 지하철 회사, 버스 회사에 보낸다. | bus\_fee, |
| FCS.UTC.023.001 | 2.3 Send | Trigger가 들어오고 c\_well==0이면 c\_well, metro\_fee를 지하철 회사, 버스 회사에 보낸다. | bus\_fee, |
| FCS.UTC.023.002 | 2.3 Send | Trigger가 들어오고 c\_well!=0이면 c\_well, | bus\_fee, |
|  |  | metro\_fee를 지하철 회사, 버스 회사에 안보낸다. | |
| FCS.UTC.024.000 | 2.4 Reset | Trigger가 들어오고 c\_well == 0이면 모든 단말기파일을 초 기화한다. | |
| FCS.UTC.024.001 | 2.4 Reset | Trigger가 들어오고 c\_well != 0이면 모든 단말기파일을 초 기화한다. | |
| FCS.UTC.024.002 | 2.4 Reset | Trigger가 들어오고 c\_well != 0이면 모든 단말기파일을 초 기화하지 않는다. | |

## 7.5 Feature pass/fail criteria

PTS, RS, FCS의 각 모듈(프로세스)은 SRA에 정의되어 있는 요구사항 (입력/ 출력, 동작) 을 모두 만족해야 한다. 각 모듈(프로세스)의 입력/ 출력 및 동작은 SRA의 Process description 항목 및 State Transition Diagram을 참조한다.

# 8 Unit test case specification

## 8.1 Test case specification identifier

### 8.1.1 Public Transportation System

**<Table 1.4 Test Case Identification>**

|  |  |  |
| --- | --- | --- |
| Test Case Identifier | Input Specification | Output Specification |
| PTS\_UTC\_1200\_000 | File!=NULL/ card\_Info-  >CID==temp\_card\_info-  >CID | Card\_info를 card\_Reader\_1.txt에 쓴다. |
| PTS\_UTC\_1200\_001 | File!=NULL/ card\_Info>CID!=temp\_card\_ info->CID | stat = InvalidInput;  card\_info의 값을 0으로 초기화시킨다. |
| PTS\_UTC\_1200\_002 | File==Null | printf("파일이 없습니다. 파일 열기 실패\n") |
| PTS\_UTC\_2111\_000 | stat==Normal | Error검사를 실행하고 가격을 책 정한다. |
| PTS\_UTC\_2111\_001 | stat!=Normal |  |
| PTS\_UTC\_2112\_000 | interval\_sec <=0 | interval\_sec += 60; |
| PTS\_UTC\_2112\_001 | state==1/interval\_sec<15/g etout==0/CRID==card\_info  ->CRID | interval\_sec += 60; stat = HopInProcessing; |
| PTS\_UTC\_2112\_002 | state==1/interval\_sec>=15 /getout==0/CRID==card\_in fo->CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_003 | state==1/interval\_sec<15/g etout!=0/CRID==card\_info-  >CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_004 | state==1/interval\_sec<15/g etout==0/CRID!=card\_info-  >CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_005 | state==1/interval\_sec<15/g etout!=0/CRID!=card\_info-  >CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_006 | state==1/interval\_sec>=15  /getout!=0/CRID==card\_inf | Stat의 값을 변경시키지 않는다. |

|  |  |  |
| --- | --- | --- |
|  | o->CRID |  |
| PTS\_UTC\_2112\_007 | state==1/interval\_sec>=15 /getout==0/CRID!=card\_inf o->CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_008 | state==1/interval\_sec>=15 /getout!=0/CRID!=card\_inf o->CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_009 | state==0/interval\_sec<15/g etout==0/CRID==card\_info  ->CRID | stat = GetOffProcessing; |
| PTS\_UTC\_2112\_010 | state==0/interval\_sec>=15 /getout==0/CRID==card\_in fo->CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_011 | state==0/interval\_sec<15/g etout!=0/CRID==card\_info-  >CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_012 | state==0/interval\_sec<15/g etout==0/CRID!=card\_info-  >CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_013 | state==0/interval\_sec<15/g etout!=0/CRID!=card\_info-  >CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_014 | state==0/interval\_sec>=15 /getout!=0/CRID==card\_inf o->CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_015 | state==0/interval\_sec>=15 /getout==0/CRID!=card\_inf o->CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2112\_016 | state==0/interval\_sec>=15 /getout!=0/CRID!=card\_inf o->CRID | Stat의 값을 변경시키지 않는다. |
| PTS\_UTC\_2113\_000 | interval\_sec <= 0 | interval\_sec += 60 \* 60 |
| PTS\_UTC\_2113\_001 | CRID < 10 | tp = 0; |
| PTS\_UTC\_2113\_002 | CRID >= 10 | tp = 1; |
| PTS\_UTC\_2113\_003 | (CRID%10) == 1 / state ==  1 / transfer == 1 / tp == 1 | price = 1650 |
| PTS\_UTC\_2113\_004 | (CRID%10) == 1 / state ==  1 / transfer == 1 / tp == 0 | price = 1550 |

|  |  |  |
| --- | --- | --- |
| PTS\_UTC\_2113\_005 | (CRID%10) == 1 / state == 1 / transfer == 1 / tp != 1  / tp != 0 | Error 출력 |
| PTS\_UTC\_2113\_006 | (CRID%10) == 1 / state ==  1 / transfer == 0 / tp == 1 | price = 1250 |
| PTS\_UTC\_2113\_007 | (CRID%10) == 1 / state ==  1 / transfer == 0 / tp == 0 | price = 1050 |
| PTS\_UTC\_2113\_008 | (CRID%10) == 1 / state == 1 / transfer == 0 / tp != 1  / tp != 0 | Error 출력 |
| PTS\_UTC\_2113\_009 | (CRID%10) == 1 / state == 1 / transfer != 0 /  transfer != 1 | Error 출력 |
| PTS\_UTC\_2113\_010 | (CRID%10) == 1 / state ==  0 / transfer == 0 / interval\_sec <= 15 / tp ==  card\_info->tp | price = 1050 |
| PTS\_UTC\_2113\_011 | (CRID%10) == 1 / state ==   1. / transfer == 0 / interval\_sec <= 15 / tp == 2. / card\_info->tp == 0 | price = 600 |
| PTS\_UTC\_2113\_012 | (CRID%10) == 1 / state ==  0 / transfer == 0 / interval\_sec <= 15 && tp  == 0 && card\_info->tp ==  1 | price = 500 |
| PTS\_UTC\_2113\_013 | (CRID%10) == 1 / state == 0 / transfer == 0 / interval\_sec <= 15 / tp !=  0 / tp != 1 / card\_info>tp != 0 / card\_info->tp !=  1 | Error 출력 |
| PTS\_UTC\_2113\_014 | (CRID%10) == 1 / state ==  0 / transfer == 0 / interval\_sec > 15 | price = 1050 |
| PTS\_UTC\_2113\_015 | (CRID%10) == 1 / state ==  0 / transfer != 0 | Error출력 |
| PTS\_UTC\_2113\_016 | (CRID%10) == 0 / state == | price = 300 |

|  |  |  |
| --- | --- | --- |
|  | 1 / transfer == 1 / tp == 1 / interval\_station == 1 || interval\_station == 4 |  |
| PTS\_UTC\_2113\_017 | (CRID%10) == 0 / state == 1 / transfer == 1 / tp == 1 / interval\_station == 2 || interval\_station == 3 | price = 600 |
| PTS\_UTC\_2113\_018 | (CRID%10) == 0 / state ==  1 / transfer == 1 / tp == 1  / interval\_station < 1 || interval\_station > 4 | price = 0 |
| PTS\_UTC\_2113\_019 | (CRID%10) == 0 / state ==  1 / transfer == 1 / tp == 0 / (interval\_sec/30) > 5 | price = 500 |
| PTS\_UTC\_2113\_020 | (CRID%10) == 0 / state ==  1 / transfer == 1 / tp == 0  / (interval\_sec/30) <= 5 | price = interval\_sec / 30 \* 100 |
| PTS\_UTC\_2113\_021 | (CRID%10) == 0 / state == 1 / transfer == 1 / tp != 0  / tp != 1 | Error 출력 |
| PTS\_UTC\_2113\_022 | (CRID%10) == 0 / state == 1 / transfer == 0 / tp == 1 / interval\_station == 1 || interval\_station == 4 | price = 0 |
| PTS\_UTC\_2113\_023 | (CRID%10) == 0 / state == 1 / transfer == 0 / tp == 1 / interval\_station == 2 || interval\_station == 3 | price = 200 |
| PTS\_UTC\_2113\_024 | (CRID%10) == 0 / state == 1 / transfer == 0 / tp == 1 / interval\_station < 1 || interval\_station > 4 | price = 0 |
| PTS\_UTC\_2113\_025 | (CRID%10) == 0 / state ==  1 / transfer == 0 / tp == 0 | price = 0 |
| PTS\_UTC\_2113\_026 | (CRID%10) == 0 / state == 1 / transfer == 0 / tp != 1  / tp != 0 | Error 출력 |
| PTS\_UTC\_2113\_027 | (CRID%10) == 0 / state != | Error 출력 |
|  | 1 |  |
| PTS\_UTC\_2113\_028 | (CRID%10)!=0/  (CRID%10) != 1 | Error 출력 |
| PTS\_UTC\_2121\_000 | stat == Normal | Short()호출 |
| PTS\_UTC\_2121\_001 | Short() / stat == Normal | Calculation()호출 |
| PTS\_UTC\_2122\_000 | (card\_info->cash-price) < 0 | stat = EShort |
| PTS\_UTC\_2122\_000 | CRID < 10 | tp = 0 |
| PTS\_UTC\_2123\_001 | CRID >= 10 | tp = 1 |
| PTS\_UTC\_2123\_002 | card\_info->transfer == 0 | card\_info->cash -= price |
| PTS\_UTC\_2123\_003 | (CRID%10) == 0 | card\_info->state = 0 |
| PTS\_UTC\_2123\_004 | (CRID%10) != 0 | card\_info->state = 1 |
| PTS\_UTC\_2130\_000 | stat == Normal | CardUpdate(),  CardReaderRecord() 호출 |
| PTS\_UTC\_2140\_000 | stat == Normal / transfer  == 1 | 0, cash 출력 |
| PTS\_UTC\_2140\_001 | stat == Normal /  transfer != 1 | price, cash 출력 |
| PTS\_UTC\_2140\_002 | stat == HopnInProcessing | Error메시지 출력 |
| PTS\_UTC\_2140\_003 | stat == GetoffProcessing | Error메시지 출력 |
| PTS\_UTC\_2140\_004 | stat == EShort | Error메시지 출력 |
| PTS\_UTC\_2140\_005 | stat == NotAddjust | Error메시지 출력 |
| PTS\_UTC\_2140\_006 | stat == InvalidInput | Error메시지 출력 |
| PTS\_UTC\_2150\_000 | file != NULL / newFile !=  NULL / feof(file) == false /  CID == fileCID | newFile에 update된 card\_info를 기록한다. |
| PTS\_UTC\_2150\_001 | file != NULL / newFile != NULL / feof(file) == false /  CID == fileCID | 기존의 카드값을 유지한다. |
| PTS\_UTC\_2160\_000 | file != NULL | card\_info를 card\_Reader\_1.txt에 쓴다. |

### 8.1.2 Recharger System

**<Table 2.4 Test Case Identification>**

|  |  |  |
| --- | --- | --- |
| **Test Case**  **Identifier** | **Input specification** | **Output specification** |
| PTS\_UTC\_120\_000 | CID=1000 / | card\_info={CID=1000, CRID=11, |
|  | if(file != NULL) | tagTime={2014, 11, 20, 19, 52, 18}, tp=1, state=1, cash=30900, transfer=0, getout=1} |
| PTS\_UTC\_120\_001 | CID=100 /  if(file != NULL) | card\_info={CID=0, CRID=0, tagTime={?, ?, ?, ?, ?, ?}, tp=0, state=0, cash=0, transfer=0, getout=0} |
| PTS\_UTC\_120\_002 | if(file == NULL) | "파일이 없습니다. 파일 열기 실패” exit(1);exit(1); |
| PTS\_UTC\_211\_000 | CID=1000 / money=20000 | card\_info.cash+=money / trigger “Update”/ trigger “Display” |
| PTS\_UTC\_211\_001 | CID=100 / money=20000 | card\_info.cash+=money / trigger “Update”/ trigger “Display” |
| PTS\_UTC\_212\_000 | card\_info={CID=1000, CRID=11, tagTime={2014, 11, 20, 19, 52, 18}, tp=1, state=1, cash=50900, transfer=0, getout=1} | fprintf(newFile, "….", card\_info->CID, …. card\_info->getout);이 한번 수행 |
| PTS\_UTC\_212\_001 | card\_info={CID=0, CRID=0,  tagTime={?, ?, ?, ?, ?, ?}, tp=0, state=0, cash=0, transfer=0, getout=0} | fprintf(newFile, "….", card\_info->CID, …. card\_info->getout);이 한번도 수행되지 못함. |
| PTS\_UTC\_212\_002 | if(file == NULL) | "파일이 없습니다. 파일 열기 실패” exit(1); |
| PTS\_UTC\_213\_000 | card\_info={CID=1000, CRID=11, tagTime={2014, 11, 20, 19, 52, 18}, tp=1, state=1, cash=50900, transfer=0, getout=1}  / money=20000 | "충전 한 시각 : 2014년 11월 21일 03시 16 분 44초”  "충전 전의 금액 : 30900원"  "충전 한 금액 : 20000원”  "충전 후의 금액 : 50900원” |
| PTS\_UTC\_213\_001 | card\_info={CID=0, CRID=0,  tagTime={?, ?, ?, ?, ?, ?}, tp=0, state=0, cash=0, transfer=0, getout=0}  / money=20000 | "충전 한 시각 : 2014년 11월 21일 03시 16 분 44초”  "충전 전의 금액 : 0원"  "충전 한 금액 : 20000원”  "충전 후의 금액 : 20000원” |

### 8.1.3 Fee Calculation System

**<Table 3.4 Test Case Identification>**

|  |  |  |
| --- | --- | --- |
| Test Case  Identifier | Input Specification | Output Specification |
| FCS.UTC.010.000 | Tick()==0/ \*.txt(단말기)!=NULL | fscanf( , ,&CardReader\_info[i].CID,,,,,,,) |
| FCS.UTC.010.001 | Tick()==0/ \*.txt(단말기)==NULL |  |
| FCS.UTC.010.002 | Tick()==1/ \*.txt(단말기)!=NULL |  |
| FCS.UTC.010.003 | Tick()==1/ \*.txt(단말기)==NULL |  |
| FCS.UTC.010.004 | CardReader\_info[i].CID==1000 | fprint(file\_CID\_1000 , ,CardReader\_info[i].CID,,,,,,,) |
| FCS.UTC.010.005 | CardReader\_info[i].CID==1001 | fprint(file\_CID\_1001, ,CardReader\_info[i].CID,,,,,,,) |
| FCS.UTC.010.006 | CardReader\_info[i].CID==1002 | fprint(file\_CID\_1002, ,CardReader\_info[i].CID,,,,,,,) |
| FCS.UTC.010.007 | CardReader\_info[i].CID==1003 | fprint(file\_CID\_1003, ,CardReader\_info[i].CID,,,,,,,) |
| FCS.UTC.010.008 | CardReader\_info[i].CID==1004 | fprint(file\_CID\_1004, ,CardReader\_info[i].CID,,,,,,,) |
| FCS.UTC.010.009 | CardReader\_info[i].CID==1005 | fprint(file\_CID\_1005, ,CardReader\_info[i].CID,,,,,,,) |
| FCS.UTC.010.010 | CardReader\_info[i].CID==1006 | fprint(file\_CID\_1006, ,CardReader\_info[i].CID,,,,,,,) |
| FCS.UTC.010.011 | CardReader\_info[i].CID>1006 ||  CardReader\_info[i].CID<1000 |  |
| FCS.UTC.021.000 | AdjustStart() | c\_well=1 |
| FCS.UTC.021.001 | !AdjustStart() | c\_well=1 |
| FCS.UTC.021.002 | CardReader\_info[i].state==1 | CardReader\_info[i].getout=1 |
| FCS.UTC.021.003 | CardReader\_info[i].state==1 | CardReader\_info[i].getout=0 |
| FCS.UTC.021.004 | CardReader\_info[i].state==0 | CardReader\_info[i].getout=1 |
| FCS.UTC.021.005 | CID\_Sort()/CID\_\*.txt !=NULL | fscanf(CID\_\*.txt, ,temp\_CardReader\_info[i].CID,,,,) |
| FCS.UTC.021.006 | CID\_Sort()/CID\_\*.txt ==NULL |  |
| FCS.UTC.021.007 | CID\_Sort()/temp\_CardReader\_info[i]!=NULL &temp\_CardReader\_info[i].tagTime작은 순 으로 | real\_CardReader\_info[i]=temp\_CardReader\_info[j] |
| FCS.UTC.021.008 | CID\_Sort()/temp\_CardReader\_info[i]==NULL  &temp\_CardReader\_info[i].tagTime작은 순  으로 |  |
| FCS.UTC.021.009 | real\_CarfReader\_info!=NULL, line!=NULL,  &bus\_fee!=NULL, &metro\_fee!=NULL | Adjust(); |
| FCS.UTC.021.010 | real\_CarfReader\_info==NULL || line==NULL  || &bus\_fee==NULL || &metro\_fee==NULL |  |
| FCS.UTC.021.011 | i>0, real\_CardReader\_info[i].price>1050 | real\_Card\_info[i].price += real\_Card\_info[i].price-  1050, |

|  |  |  |
| --- | --- | --- |
|  |  | real\_Card\_info[i].price-=1050. |
| FCS.UTC.021.012 | i<=0 || real\_CardReader\_info[i].price<=1050 | real\_Card\_info[i].price += real\_Card\_info[i].price-  1050, real\_Card\_info[i].price-=1050. |
| FCS.UTC.021.013 | Index=size-1; for(i=size-1;i>=0;i--) if((real\_card\_info[i].state==1)&& (real\_card\_info[i].transfer==0)) | index=i-1; |
| FCS.UTC.021.014 | Index=size-1; for(i=size-1;i>=0;i--) if((real\_card\_info[i].state!=1)|| (real\_card\_info[i].transfer!=0)) | . |
| FCS.UTC.021.016 | Index=size-1; for(i=size-1;i>=0;i--)  {  if((real\_card\_info[i].state==1)&&  (real\_card\_info[i].transfer==0))  { for(j=index;j>=i;j--){ | real\_fee+=real\_card\_info[j].price |
| FCS.UTC.021.017 | Index==NULL | real\_fee+= real\_CardReader\_info[i].price |
| FCS.UTC.021.018 | real\_CardReader\_info[i-1].tp != real\_CardReader\_info[i].tp | total\_fee+=real\_fee |
| FCS.UTC.021.019 | real\_CardReader\_info[i-1].tp == real\_CardReader\_info[i].tp | total\_fee+=real\_fee |
| FCS.UTC.021.020 | Index=size-1; for(i=size-1;i>=0;i--)  {  if((real\_card\_info[i].state==1)&&  (real\_card\_info[i].transfer==0))  { for(j=index;j>=i;j--){ | temp\_fee+=real\_CardReader\_info[i].price |
| FCS.UTC.021.021 | index==NULL | temp\_fee+=real\_CardReader\_info[i].price |
| FCS.UTC.021.022 | if(j==i || (real\_card\_info[j-1].tp !=real\_card\_info[j].tp)){ if(real\_card\_info[j].tp==0) | \*bus\_fee+=temp\_fee/total\_fee\*real\_fee |
| FCS.UTC.021.023 | if(j!=i || (real\_card\_info[j-1].tp ==real\_card\_info[j].tp)){ if(real\_card\_info[j].tp==0) | \*bus\_fee+=temp\_fee/total\_fee\*real\_fee |

|  |  |  |
| --- | --- | --- |
| FCS.UTC.021.024 | if(j==i || (real\_card\_info[j-1].tp !=real\_card\_info[j].tp)){ if(real\_card\_info[j].tp==1) | Adjust상태 일 때, ((real\_CardReader\_info[i-  1].tp !=real\_CardReader\_info[i].tp) || i==0), real\_CardReader\_info[i].tp==1이면  \*metro\_fee+=temp\_fee/total\_fee\*real\_fee를 수행 한다. |
| FCS.UTC.021.025 | if(j!=i || (real\_card\_info[j-1].tp ==real\_card\_info[j].tp)){ if(real\_card\_info[j].tp==1) | \*metro\_fee+=temp\_fee/total\_fee\*real\_fee |
| FCS.UTC.021.026 | Index=size-1; for(i=size-1;i>=0;i--)  {  if((real\_card\_info[i].state==1)&&  (real\_card\_info[i].transfer==0))  { for(j=index; j>=i; j--){}  } | index=i-1, total\_feee=0, real\_fee=0, temp\_fee=0 |
| FCS.UTC.021.027 | If문 끝난 다음이 아닐때 | index=i-1, total\_feee=0, real\_fee=0, temp\_fee=0 |
| FCS.UTC.021.028 | bus\_fee!=NULL, metro\_fee!=NULL | c\_well=0 |
| FCS.UTC.021.029 | bus\_fee==NULL, metro\_fee==NULL | c\_well=0 |
| FCS.UTC.021.029 | c\_well==0 | enable Display. |
| FCS.UTC.021.030 | c\_well!=0 | enable Display. |
| FCS.UTC.021.031 | c\_well==0 | Trigger Send |
| FCS.UTC.021.032 | c\_well!=0 | Trigger Send |
| FCS.UTC.021.033 | c\_well==0 | Trigger Reset |
| FCS.UTC.021.034 | c\_well!=0 | Trigger Reset |
| FCS.UTC.022.000 | Enable/c\_well==0 | printf("버스 정산 금액: %d\n",bus\_fee); printf("지하철 정산 금액: %d\n",metro\_fee); |
| FCS.UTC.022.001 | Enable/c\_well==0 | printf("버스 정산 금액: %d\n",bus\_fee); printf("지하철 정산 금액: %d\n",metro\_fee); |
| FCS.UTC.022.002 | c\_well!=0 |  |
| FCS.UTC.023.000 | Trigger/c\_well==0 | file\_bus=fopen("send\_bus.txt","w"); fprintf(file\_bus,"%d\n",c\_well); fprintf(file\_bus,"정산금액: %d\n",bus\_fee);    file\_metro=fopen("send\_metro.txt","w"); fprintf(file\_metro,"%d\n",c\_well); fprintf(file\_metro,"총 정산금액: %d",metro\_fee); |
| FCS.UTC.023.001 | Trigger/c\_well==0 | file\_bus=fopen("send\_bus.txt","w"); fprintf(file\_bus,"%d\n",c\_well); fprintf(file\_bus,"정산금액: %d\n",bus\_fee);    file\_metro=fopen("send\_metro.txt","w"); fprintf(file\_metro,"%d\n",c\_well); fprintf(file\_metro,"총 정산금액: %d",metro\_fee); |
| FCS.UTC.023.002 | c\_well!=0 |  |
| FCS.UTC.024.000 | Trigger/c\_well==0 | remove(fname); rename("temp.txt",fname); |
| FCS.UTC.024.001 | Trigger/c\_well==0 | remove(fname); rename("temp.txt",fname); |
| FCS.UTC.024.002 | c\_well!=0 |  |

## 8.2 Test items

### 8.2.1 Public Transportation System

**<Table 1.3 Test Design Identification>참조**

### 8.2.2 Recharger System

**<Table 2.3 Test Design Identification>참조**

### 8.2.3 Fee Calculation System

**<Table 3.3 Test Design Identification>참조**

## 8.3 Input specifications

### 8.3.1 Public Transportation System

**<Table 1.4 Test Case Identification>참조**

### 8.3.2 Recharger System

**<Table 2.4 Test Case Identification>참조**

### 8.3.3 Fee Calculation System

**<Table 3.4 Test Case Identification>참조**

## 8.4 Output specifications

### 8.4.1 Public Transportation System

**<Table 1.4 Test Case Identification>참조**

### 8.4.2 Recharger System

**<Table 2.4 Test Case Identification>참조**

### 8.4.3 Fee Calculation System

**<Table 3.4 Test Case Identification>참조**

# 9 Testing tasks

**<Table 5 Testing tasks & Schedule>**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task | Predecessor tasks | Special skills | Effort | Finish  Date |
| Unit Test Plan 작성 | T3.2014.PTS.SRS 작성  T3.2014.RGS.SRS 작성  T3.2014.FCS.SRS 작성  PTS,RS,FCS 구현 |  | 3 |  |
| Test Design Specification | Task1 | PTS,RS,FCS에 대한 이해 | 5 |  |
| Test Case Specification | Task2 | PTS,RS,FCS에 대한 이해 | 5 |  |
| Test Execution | Task3 | Test Code 작성 | 4 |  |
| Test result report | Task4 |  | 2 |  |

# 10 Environmental needs

C Language Compiler/Linker Ex)Visual Studio 2010

# 11 Unit Test deliverables

# 12 Schedules

<Table 5 Testing tasks & Schedule>참조