Unit Testing Report

for Network Printer System

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

Project Team

T1 Team

Date

2015-11-09

Team Information

하 지 용(201011375)

유 치 영(201211823)

허 인 석(201213220)

라 가 영(201214262)

Table of Contents

1 Introduction

1.1 Objectives

본 문서는 2015년도 2학기 소프트웨어 공학 개론 수업의 T1 Team이 개발한 Network Print System(이하 NPS)을 수행한 결과에 대한 Report 문서이다. Test 요소들에 대한 Test Case와 Test 수행 결과에 대한 내용을 담고 있다.

1.2 Reference

DS-2015SE-NPS-SRS-1.0

T1-2015-NPS-SRA-1.2

T1-2015-NPS-SDA-1.0

T1-2015-NPS-UTP-1.0

2 Unit test case specification

2.1 Test case specification identifier

<Table 1.1 Test Case Identification>

Identifier	Feature	Pass / Fail Criteria
NPS.UTC.1000	입력된 명령을 분류하여 각 명령을 Admin Sub System과 Print Sub System으로 전달한다.	
NPS.UTC.1000.00	user add user a1	ERROR:not supported command
NPS.UTC.1000.01	user del user a1	ERROR:not supported command
NPS.UTC.1000.02	user show user list	ERROR:not supported command
NPS.UTC.2031	관리자가 유저를 추가/삭제 및 유저리스트 확인을 할 것인지를 Add User, Delete User, Show User List로 분류하여 각 프로세스를 실행한다	
NPS.UTC.2031.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외	
NPS.UTC.2032	유저를 추가한다.	
NPS.UTC.2032.00	AddUser("a1")	g_UserInfoList={"a1"}

NPS.UTC.2032.01	g_UserInfoList={"a1"} AddUser("a1")	ERROR:a1 is Already registed.
NPS.UTC.2032.02	AddUser("a11111111111111111111111111111111111111111111111")	ERROR:Incorrect UserName
NPS.UTC.2032.03	AddUser(" ")	ERROR:Incorrect UserName
NPS.UTC.2032.04	AddUser("a1 a2")	g_UserInfoList={"a1 a2"}
NPS.UTC.2032.05	AddUser("&a1")	g_UserInfoList={"&a1"}
NPS.UTC.2033	유저를 삭제한다.	
NPS.UTC.2033.00	g_UserInfoList={"a1"} DelUser("a1")	g_UserInfoList==0
NPS.UTC.2033.01	g_UserInfoList={"a1"} DelUser("a2")	ERROR:a2 is not registered
NPS.UTC.2034	유저 리스트를 확인한다.	
NPS.UTC.2034.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외	
NPS.UTC.2041	유저가 Ink를 리필 할 것인지, Paper를 리필 할 것인지Supplies Data를 Ink와 Paper로 분류하여 각 프로세스를 실행한다.	
NPS.UTC.2041.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외	
NPS.UTC.2042	유저가 Ink를 리필 할 경우, Supplies Data 내 충전할 Ink량을 수정한다.	
NPS.UTC.2042.00	KEYWORD_INK == TRUE inkValue =100	g_SuppliesData.Ink={100}
NPS.UTC.2042.01	KEYWORD_INK == TRUE inkValue = -100	ERROR:Invalid Input ink Value
NPS.UTC.2043	유저가 Paper를 리필 할 경우, Supplies Data 내 충전할 Paper량을 수정한다.	
NPS.UTC.2043.00	KEYWORD_PAPER == TRUE paperValue =100	g_SuppliesData.Paper={100}
NPS.UTC.2043.01	KEYWORD_PAPER == TRUE paperValue = -100	ERROR:Invalid Input paper Value
NPS.UTC.2043.02	KEYWORD_PAPER == TRUE paperValue = 100 KEYWORD_INK == TRUE inkValue = 100	g_SuppliesData.Paper={100} g_SuppliesData.Ink={100}
NPS.UTC.2043.03	KEYWORD_PAPER == TRUE	g_SuppliesData.Paper={

	paperValue = 100 KEYWORD_INK == TRUE inkValue = -100	100} ERROR:Invalid Input ink Value
NPS.UTC.2043.04	KEYWORD_PAPER == TRUE paperValue = -100 KEYWORD_INK == TRUE inkValue =100	ERROR:Invalid Input paper Value g_SuppliesData.Ink={100}
NPS.UTC.2043.05	KEYWORD_PAPER == TRUE paperValue = -100 KEYWORD_INK == TRUE inkValue <= 0	"ERROR:Invalid Input paper Value ERROR:Invalid Input ink Value
NPS.UTC.2043.06	KEYWORD_INK == TRUE inkValue =100 KEYWORD_PAPER == TRUE paperValue =100	g_SuppliesData.Ink={100} g_SuppliesData.Paper={100}
NPS.UTC.2043.07	KEYWORD_INK == TRUE inkValue =100 KEYWORD_PAPER == TRUE paperValue = -100	g_SuppliesData.Ink={100} ERROR:Invalid Input paper Value
NPS.UTC.2043.08	KEYWORD_INK == TRUE inkValue = -100 KEYWORD_PAPER == TRUE paperValue =100	ERROR:Invalid Input ink Value g_SuppliesData.Paper={100}
NPS.UTC.2043.09	KEYWORD_INK == TRUE inkValue = -100 KEYWORD_PAPER == TRUE paperValue = -100	ERROR:Invalid Input ink Value ERROR:Invalid Input paper Value
NPS.UTC.3010	명령이 추가 / 삭제, 전체 취소 인지를 확인하며 Add Job, Delete Job, Cancel All로 분류하여 각 프로세스를 실행한다.	
NPS.UTC.3010.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외	
NPS.UTC.3020	Admin Sub System에서 authorize check가 된 인쇄 추가 명령을 받아 Job Wait List과 Print Context를 수정한다.	
NPS.UTC.3020.00	AddJob(userIdx, "test.txt");	g_JobWaitList={test.txt}
NPS.UTC.3020.01	AddJob(userIdx1, "test.txt"); AddJob(userIdx2, "test.txt");	g_JobWaitList={test.txt1, test.txt2}

NPS.UTC.3020.02	GetNodeCount(g_JobWaitList)==5 AddJob(userIdx, "test.txt");	ERROR:JOB WAIT LIST FULL
NPS.UTC.3020.03	AddJob(userIdx, "test.txt"); test.txt != EXIST	ERROR:request file open failed
NPS.UTC.3030	Admin Sub System에서 인쇄 삭제 명령을 받아 Job Wait List과 Print Context를 수정한다	
NPS.UTC.3030.00	g_JobWaitList={(1, test1.txt), (2, test2.txt)} DeleteJob(userIdx1);	g_JobWaitList={test2.txt }
NPS.UTC.3030.01	GetNodeCount(g_JobWaitList)==0 DeleteJob(userIdx);	ERROR: UID : 0 Job not found
NPS.UTC.3030.02	g_JobWaitList={(1, test1.txt)} DelelteJob(1)	g_PrinterContext.pCurrentJob=Printing g_PrinterContext.pCurrentJob=NULL
NPS.UTC.3030.03	g_JobWaitList={(1, test1.txt), (2, test2.txt)} DelelteJob(1);	g_JobWaitList={test2.txt }
NPS.UTC.3030.04	g_PrinterContext.status == Printing DeleteJob(currentUserIdx);	g_PrinterContext.status =print g_PrinterContext.status =wait
NPS.UTS.3040	Command Parser에서 Cancel All Command를 받아 Job Wait List과 Print Context를 수정한다.	
NPS.UTS.3040 00	g_JobWaitList={(1, test1.txt), (2, test2.txt), (3, test3.txt)} CancelAll();	g_PrinterContext.status = CancelSignal DeleteNode(&g_JobWaitList, pJobNode)
NPS.UTC.3050	JOB WAIT LIST와 PRINTER CONTEXT의 데이터를 받아와 각 프로세스로 전달한다	
NPS.UTC.3050.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외	
NPS.UTC.3060	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받아 대기중인 상태를 LCD에 표시하고 PRINTER CONTEXT에 Status를 Wait으로 변경한다.	
NPS.UTC.3060.00	Interfaces는 testing 제외	
NPS.UTC.3070	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받아 취소 상태를 LCD에 표시하고 PRINTER CONTEXT에 Status를 Wait로 변경한다.	

NPS.UTC.3070.00	pCurrentJob != NULL	g_PrinterContext.pCurrentJob = NULL g_PrinterContext.status = Wait
NPS.UTC.3070.01	pCurrentJob = NULL	g_PrinterContext.status = Wait
NPS.UTC.3080	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받아 충전이 요구되는 상태를 LCD에 표시하고 PRINTER CONTEXT에 Status를 Need Supplies로 변경한다.	
NPS.UTC.3080.00	Interfaces는 testing 제외	
NPS.UTC.3090	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받고, JOB WAIT LIST에서 최상위 작업을 불러와 출력을 시작한다. 출력 중인 상태를 LCD에 표시하고 PRINTER CONTEXT에 Status를 Printing으로 변경한다	
NPS.UTC.3090.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외	
NPS.UTC.3100	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받고, 출력중인 작업을 지속한다.	
NPS.UTC.3100.00	pCurrentJob.needPaper != pCurrentJob.printedCnt !feof(pCurrentJob.plnFile) == TRUE	g_PrinterContext.status = Printing Printing 출력
NPS.UTC.3100.01	pCurrentJob.needPaper != pCurrentJob.printedCnt !feof(pCurrentJob.plnFile) == FALSE	g_PrinterContext.status = Wait Wait 출력
NPS.UTC.3100.02	pCurrentJob.needPaper == pCurrentJob.printedCnt	fclose(pCurrentJob.plnFile) fclose(pCurrentJob.pOutFile) free(pCurrentJob) g_PrinterContext.pCurrentJob = NULL g_PrinterContext.status = Wait Wait 출력
NPS.UTC.3110	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받고, 잉크와 용지를 SUPPLIES DATA에서	

	Paper가 0이 아닐 때 10만큼, Ink가 0이 아닐 때 100만큼 감소 후 같은 량을PRINTER CONEXT의 Ink와 Paper에 증가시킨다.	
NPS.UTC.3110.00	g_PrinterContext.ink + refillInk > MAX_SUPPLIES_INK	refillInk = refillInk – (g_PrinterContext.ink + refillInk_MAX_SUPPLIES_INK)
NPS.UTC.3110.01	g_PinterContext.paper + refillPaper > MAX_SUPPLIES_PAPER	refillPaper = refillPaper – (g_PrinterContext.paper + refillPaper – MAX_SUPPLIES_PAPAER)
NPS.UTC.3110.02	(g_SuppliesData.Ink != 0 && g_PrinterContext.ink < MAX_SUPPLIES_INK) || (g_SuppliesData.Paper != 0 && g_PrinterContext.paper < MAX_SUPPLIES_PAPER)	g_PrinterContext.status = Refilling
NPS.UTC.3110.03	(g_SuppliesData.Ink = 0 || g_PrinterContext.ink > MAX_SUPPLIES_INK) && (g_SuppliesData.Paper = 0 || g_PrinterContext.paper > MAX_SUPPLIES_PAPER)	g_PrinterContext.status = Wait

2.2 Test items

<Table 1.1 Test Case Identification> 참조

2.3 Input specifications

<Table 1.1 Test Case Identification> 참조

2.4 Output specifications

<Table 1.1 Test Case Identification> 참조

3 Environmental needs

T1-2015-NPS-UTP-1.0 Environmental needs 항목 참조

4 Unit test summary report

4.1 Test summary report identifier

Identifier	Feature	Pass / Fail Criteria	Test Result
NPS.UTC.1000	입력된 명령을 분류하여 각 명령을 Admin Sub System과 Print Sub System으로 전달한다.		
NPS.UTC.1000.00	user add user a1	ERROR:not supported command	PASSED
NPS.UTC.1000.01	user del user a1	ERROR:not supported command	PASSED
NPS.UTC.1000.02	user show user list	ERROR:not supported command	PASSED
NPS.UTC.2031	관리자가 유저를 추가/삭제 및 유저리스트 확인을 할 것인지를 Add User, Delete User, Show User List로 분류하여 각 프로세스를 실행한다		
NPS.UTC.2031.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외		PASSED
NPS.UTC.2032	유저를 추가한다.		
NPS.UTC.2032.00	AddUser("a1")	g_UserInfoList={"a1"}	PASSED
NPS.UTC.2032.01	g_UserInfoList={"a1"} AddUser("a1")	ERROR:a1 is Already registed.	PASSED
NPS.UTC.2032.02	AddUser("a11111111111111111111111111111111111111")	ERROR:Incorrect UserName	PASSED
NPS.UTC.2032.03	AddUser(" ")	ERROR:Incorrect UserName	PASSED
NPS.UTC.2032.04	AddUser("a1 a2")	g_UserInfoList={"a1 a2"}	PASSED
NPS.UTC.2032.05	AddUser("&a1")	g_UserInfoList={"&a1"}	PASSED
NPS.UTC.2033	유저를 삭제한다.		
NPS.UTC.2033.00	g_UserInfoList={"a1"} DelUser("a1")	g_UserInfoList==0	PASSED

NPS.UTC.2033.01	g_UserInfoList={"a1"} DelUser("a2")	ERROR:a2 is not registered	PASSED
NPS.UTC.2034	유저 리스트를 확인한다.		
NPS.UTC.2034.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외		PASSED
NPS.UTC.2041	유저가 Ink를 리필 할 것인지, Paper를 리필 할 것인지Supplies Data를 Ink와 Paper로 분류하여 각 프로세스를 실행한다.		
NPS.UTC.2041.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외		PASSED
NPS.UTC.2042	유저가 Ink를 리필 할 경우, Supplies Data 내 충전할 Ink량을 수정한다.		
NPS.UTC.2042.00	KEYWORD_INK == TRUE inkValue =100	g_SuppliesData.Ink ={100}	PASSED
NPS.UTC.2042.01	KEYWORD_INK == TRUE inkValue = -100	ERROR:Invalid Input ink Value	PASSED
NPS.UTC.2043	유저가 Paper를 리필 할 경우, Supplies Data 내 충전할 Paper량을 수정한다.		
NPS.UTC.2043.00	KEYWORD_PAPER == TRUE paperValue =100	g_SuppliesData.Paper={100}	PASSED
NPS.UTC.2043.01	KEYWORD_PAPER == TRUE paperValue = -100	ERROR:Invalid Input paper Value	PASSED
NPS.UTC.2043.02	KEYWORD_PAPER == TRUE paperValue = 100 KEYWORD_INK == TRUE inkValue = 100	g_SuppliesData.Paper={100} g_SuppliesData.Ink ={100}	PASSED
NPS.UTC.2043.03	KEYWORD_PAPER == TRUE paperValue = 100 KEYWORD_INK == TRUE inkValue = -100	g_SuppliesData.Paper={100} ERROR:Invalid Input ink Value	PASSED
NPS.UTC.2043.04	KEYWORD_PAPER == TRUE paperValue = -100 KEYWORD_INK == TRUE inkValue =100	ERROR:Invalid Input paper Value g_SuppliesData.Ink ={100}	PASSED
NPS.UTC.2043.05	KEYWORD_PAPER == TRUE	"ERROR:Invalid	PASSED

	paperValue = -100 KEYWORD_INK == TRUE inkValue <= 0	Input paper Value ERROR:Invalid Input ink Value	
NPS.UTC.2043.06	KEYWORD_INK == TRUE inkValue =100 KEYWORD_PAPER == TRUE paperValue =100	g_SuppliesData.Ink ={100} g_SuppliesData.Pap er={100}	PASSED
NPS.UTC.2043.07	KEYWORD_INK == TRUE inkValue =100 KEYWORD_PAPER == TRUE paperValue = -100	g_SuppliesData.Ink ={100} ERROR:Invalid Input paper Value	PASSED
NPS.UTC.2043.08	KEYWORD_INK == TRUE inkValue = -100 KEYWORD_PAPER == TRUE paperValue =100	ERROR:Invalid Input ink Value g_SuppliesData.Pap er={100}	PASSED
NPS.UTC.2043.09	KEYWORD_INK == TRUE inkValue = -100 KEYWORD_PAPER == TRUE paperValue = -100	ERROR:Invalid Input ink Value ERROR:Invalid Input paper Value	PASSED
NPS.UTC.3010	명령이 추가 / 삭제, 전체 취소 인지를 확인하며 Add Job, Delete Job, Cancel All로 분류하여 각 프로세스를 실행한다.		
NPS.UTC.3010.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외		PASSED
NPS.UTC.3020	Admin Sub System에서 authorize check가 된 인쇄 추가 명령을 받아 Job Wait List과 Print Context를 수정한다.		
NPS.UTC.3020.00	AddJob(userIdx, "test.txt");	g_JobWaitList={test .txt}	PASSED
NPS.UTC.3020.01	AddJob(userIdx1, "test.txt"); AddJob(userIdx2, "test.txt");	g_JobWaitList={test .txt1,test.txt2}	PASSED
NPS.UTC.3020.02	GetNodeCount(g_JobWaitList)==5 AddJob(userIdx, "test.txt");	ERROR:JOB WAIT LIST FULL	PASSED
NPS.UTC.3020.03	AddJob(userIdx, "test.txt"); test.txt != EXIST	ERROR:request file open failed	PASSED

NPS.UTC.3030	Admin Sub System에서 인쇄 삭제 명령을 받아 Job Wait List과 Print Context를 수정한다		
NPS.UTC.3030.00	g_JobWaitList={(1, test1.txt), (2, test2.txt)} DeleteJob(userIdx1);	g_JobWaitList={test2.txt}	PASSED
NPS.UTC.3030.01	GetNodeCount(g_JobWaitList)==0 DeleteJob(userIdx);	ERROR: UID : 0 Job not found	PASSED
NPS.UTC.3030.02	g_JobWaitList={(1, test1.txt)} DeleIteJob(1)	g_PrinterContext.pCurrentJob=Printing g_PrinterContext.pCurrentJob=NULL	PASSED
NPS.UTC.3030.03	g_JobWaitList={(1, test1.txt), (2, test2.txt)} DeleIteJob(1);	g_JobWaitList={test2.txt}	PASSED
NPS.UTC.3030.04	g_PrinterContext.status == Printing DeleteJob(currentUserIdx);	g_PrinterContext.status=print g_PrinterContext.status=wait	PASSED
NPS.UTS.3040	Command Parser에서 Cancel All Command를 받아 Job Wait List과 Print Context를 수정한다.		
NPS.UTS.3040 00	g_JobWaitList={(1, test1.txt), (2, test2.txt), (3, test3.txt)} CancelAll();	g_PrinterContext.status = CancelSignal DeleteNode(&g_JobWaitList, pJobNode)	PASSED
NPS.UTC.3050	JOB WAIT LIST와 PRINTER CONTEXT의 데이터를 받아와 각 프로세스로 전달한다		
NPS.UTC.3050.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외		PASSED
NPS.UTC.3060	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받아 대기중인 상태를 LCD에 표시하고 PRINTER CONTEXT에		

	Status를 Wait으로 변경한다.		
NPS.UTC.3060.00	Interfaces는 testing 제외		PASSED
NPS.UTC.3070	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받아 취소 상태를 LCD에 표시하고 PRINTER CONTEXT에 Status를 Wait로 변경한다.		
NPS.UTC.3070.00	pCurrentJob != NULL	g_PrinterContext.pCurrentJob = NULL g_PrinterContext.status = Wait	PASSED
NPS.UTC.3070.01	pCurrentJob = NULL	g_PrinterContext.status = Wait	PASSED
NPS.UTC.3080	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받아 충전이 요구되는 상태를 LCD에 표시하고 PRINTER CONTEXT에 Status를 Need Supplies로 변경한다.		
NPS.UTC.3080.00	Interfaces는 testing 제외		PASSED
NPS.UTC.3090	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받고, JOB WAIT LIST에서 최상위 작업을 불러와 출력을 시작한다. 출력 중인 상태를 LCD에 표시하고 PRINTER CONTEXT에 Status를 Printing으로 변경한다		
NPS.UTC.3090.00	전달 역할 등 단순한 프로세스 및 UI구성을 담당하는 프로세스는 테스트에서 제외		PASSED
NPS.UTC.3100	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받고, 출력중인 작업을 지속한다.		
NPS.UTC.3100.00	pCurrentJob.needPaper != pCurrentJob.printedCnt !feof(pCurrentJob.plnFile) == TRUE	g_PrinterContext.status = Printing Printing 출력	PASSED
NPS.UTC.3100.01	pCurrentJob.needPaper != pCurrentJob.printedCnt !feof(pCurrentJob.plnFile) == FALSE	g_PrinterContext.status = Wait Wait 출력	PASSED
NPS.UTC.3100.02	pCurrentJob.needPaper == pCurrentJob.printedCnt	fclose(pCurrentJob.plnFile)	PASSED

		fclose(pCurrentJob.pOutFile) free(pCurrentJob) g_PrinterContext.pCurrentJob = NULL g_PrinterContext.status = Wait Wait 출력	
NPS.UTC.3110	JOB WAIT LIST와 PRINTER CONTEXT를 Trigger로 전달받고, 잉크와 용지를 SUPPLIES DATA에서 Paper가 0이 아닐 때 10만큼, Ink가 0이 아닐 때 100만큼 감소 후 같은 량을PRINTER CONEXT의 Ink와 Paper에 증가시킨다.		
NPS.UTC.3110.00	g_PrinterContext.ink + refillInk > MAX_SUPPLIES_INK	refillInk = refillInk – (g_PrinterContext.ink + refillInk_MAX_SUPPLIES_INK)	PASSED
NPS.UTC.3110.01	g_PinterContext.paper + refillPaper > MAX_SUPPLIES_PAPER	refillPaper = refillPaper – (g_PrinterContext.paper + refillPaper – MAX_SUPPLIES_PAPAER)	PASSED
NPS.UTC.3110.02	(g_SuppliesData.Ink != 0 && g_PrinterContext.ink < MAX_SUPPLIES_INK) || (g_SuppliesData.Paper != 0 && g_PrinterContext.paper < MAX_SUPPLIES_PAPER)	g_PrinterContext.status = Refilling	PASSED
NPS.UTC.3110.03	(g_SuppliesData.Ink = 0 || g_PrinterContext.ink > MAX_SUPPLIES_INK) && (g_SuppliesData.Paper = 0 || g_PrinterContext.paper > MAX_SUPPLIES_PAPER)	g_PrinterContext.status = Wait	PASSED

4.2 Evaluation

Total test case : 50개

Passed : 50개

Failed : 0개