



국민대학교
전자정보통신대학
컴퓨터공학부

CONFIDENTIALITY/SECURITY WARNING

이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 캡스톤 디자인 I 수강 학생 중 프로젝트 "SSD Test 자동화 시스템"을 수행하는 팀 "1g's dormitory"의 팀원들의 자산입니다. 국민대학교 컴퓨터공학부 및 팀 "1g's dormitory"의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

캡스톤 디자인 I 종합설계 프로젝트

| | |
|--------|------------------|
| 프로젝트 명 | SSD Test 자동화 시스템 |
| 팀 명 | 1g's dormitory |
| 문서 제목 | 계획서 |

| | |
|---------|------------|
| Version | 1.1 |
| Date | 2018-MAR-8 |

| | |
|----|-----|
| 이름 | 이진식 |
|----|-----|

| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |

문서 정보 / 수정 내역

| 수정날짜 | 수정자 | Revision | 추가/수정 항목 | 내 용 |
|------------|-----|----------|----------|----------------------|
| 2018-03-07 | 이진식 | 1.0 | 최초 작성 | 최초 작성 |
| 2018-03-08 | 이진식 | 1.1 | 2.2절 | 개발 내용을 part별로 분리, 서술 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |

목 차

| | | |
|----------|----------------------|-----------|
| 1 | 개요 | 4 |
| 1.1 | 프로젝트 개요 | 4 |
| 1.2 | 추진 배경 및 필요성 | 4 |
| 2 | 개발 목표 및 내용 | 5 |
| 2.1 | 목표 | 5 |
| 2.2 | 연구/개발 내용 | 5 |
| 2.2.1 | Test System Server | 5 |
| 2.2.2 | Host Server | 6 |
| 2.2.3 | Web | 6 |
| 2.3 | 개발 결과 | 7 |
| 2.3.1 | 개발 결과 시나리오 | 7 |
| 2.3.2 | Use Case Diagram | 7 |
| 2.3.3 | 시스템 설계도 | 8 |
| 2.4 | 기대효과 및 활용방안 | 8 |
| 3 | 배경 기술 | 9 |
| 3.1 | 기술적 요구사항 | 9 |
| 3.1.1 | 프로젝트 개발환경 | 9 |
| 3.2 | 현실적 제한 요소 및 그 해결 방안 | 9 |
| 3.2.1 | 소프트웨어 | 9 |
| 4 | 개발 일정 및 자원 관리 | 10 |
| 4.1 | 개발 일정 | 10 |
| 4.2 | 일정 별 주요 산출물 | 10 |

| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |

1 개요

1.1 프로젝트 개요

최근 저장매체 시장이 HDD에서 SSD로 바뀌고 있다. SSD가 HDD에 비하여 속도가 빠르고, 전력 소비가 적고, 소음이 적기때문에 SSD시장이 더욱 활성화될 것으로 예상된다. 이에 따라서 여러 회사에서 SSD 제품을 출시하고 있고, Fadu에서는 현재 SSD를 개발하고 있기 때문에 현재 개발중인 제품들을 검증하기 위한 시스템이 필수적이다. 현재 나와 있는 Test Tool들은 대부분 local machine에서 Test하기 때문에 대량으로 Test하기에 비효율적이다. Fadu에서는 이러한 기존 Tool들의 한계점을 보완하기 위해, 새로운 Test System을 산학과제로 다루기로 했다.

1.2 추진 배경 및 필요성

1.2.1 기존에 개발된 시스템 현황 및 한계점

현재는 SSD Test를 할 때, Local Machine에서 SSD Benchmark Tool을 실행하기 때문에 비용(인력과 시간)이 크다. 또한 Test 결과를 Standard Out으로 보고 Log File들을 Dump file로 Database에 저장하기 때문에 Firmware/Test Script 별 SSD Test history management가 어렵다.

1.2.2 개발할 시스템의 필요성

현재 사내의 Test 방식으로는 여러 대의 SSD를 Test해보고 그에 따른 결과를 받아보고 싶어한다. 하지만 Local Machine으로 테스트하는 방식은 1:1방식이기 때문에 인력/시간이 많이 필요로 한다. Firmware와 test script 역시 1:1 방식에서는 version control 등이 어려울 수 있다. 또한 Firmware, test script별 테스트 이력관리가 어려우며, 회사에 와서 Test를 하는 방법이 아니라면 Test를 할 방법이 없다. 따라서 기존의 SSD Test System의 비효율성을 보완하여 더 많은 SSD를 빠르고 편리하게 Test해 줄 수 있는 System이 필요하다.

| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |

2 개발 목표 및 내용

2.1 목표

기존에 있는 SSD Test System에 대한 구조와 이해를 바탕으로 보다 효율적인 Test System을 개발한다.

이를 위한 주요 목표는 다음과 같다.

1. Remote로 Test가 가능한 시스템
원격으로 SSD Test가 실행되고 log file과 결과값을 원격으로 받아볼 수 있어야 한다.
2. 멀티로 Test가 가능한 시스템
동시에 여러 대의 SSD test server를 동작 시키고 비동기식으로 결과값을 받아볼 수 있어야 한다.
3. 시각적으로 보이는 웹
Test 상황과 test결과등을 시각적으로 받아볼 수 있어야 한다(pass-fail 등).
4. Firmware, script별 test 이력관리
Firmware와 Test script별로 test이력을 관리한다.
(ex. FW version : 1.0, Test script version : 1.0, Test 결과 : Pass)

위의 3가지 기능을 하는 SSD Test System을 개발한다.

2.2 연구/개발 내용

2.2.1 Test System Server

Test System Server는 실제 SSD에 연결되어 있는 PC위에 올라갈 서버이다. Host 서버에서 요청한 명령에 따라 오픈소스 SSD test tool, 자체개발 SSD test tool들을 실행시켜 그 결과값을 Host 서버에 전송하는 역할을 한다.

1. Local PC 내에서의 SSD test tool을 리서치 한 후 SSD test tool의 I/O를 정리한다.
2. SSD test tool의 데이터를 네트워크로 전송 가능한 message 형태로 가공한다.
3. Host server와 네트워크로 연결한다.
4. Host server에서 오는 요청을 받아 test tool을 실행시킬 수 있는 기능을 추가한다. 그에 따른 test result log를 Host Server에 전송하는 기능을 추가한다.
5. Host server에서 오는 요청을 받아 Firmware를 업데이트 할 수 있는 기능을 추가한다. 그에 따른 Firmware Update log를 Host Server에 전송하는 기능을 추가한다.
6. Host server에서 오는 요청을 받아 test script를 실행시킬 수 있는 기능을 추가한다. 그에 따른 test result log를 Host Server에 전송하는 기능을 추가한다.

| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |

7. 현재 Centos 6.9버전으로 fix 되어있는 OS를 다양하게 지원할 수 있는 서버를 개발한다.

2.2.2 Host Server

Host서버는 Test System Server에 internet상에서 오는 Test 명령과 Test script, Firmware를 전송해 주고 그에 따른 결과값을 DB에 저장해주는 역할을 하는 Server이다. 또한 Web상에 Test 상황과 Test 이력들을 전송해주는 역할을 한다.

1. Test System server로의 기본적인 통신(GET, POST등)을 할 수 있는 서버를 개발한다.
2. Test System server로의 명령(test start등)네트워크로 전송 가능한 message 형태로 가공한다.
3. Test System server로의 명령(test start)네트워크로 전송하여 test를 실행한다.
4. Test System server에서 전송한 test result log들을 database에 저장한다.
5. 여러 대의 Test System server로의 명령(test start)네트워크로 전송하여 여러 대의 PC에서 test를 실행한다.
6. Test System server로의 명령(Firmware update)네트워크로 전송하여 Firmware update를 실행한다.
7. Test System server에서 전송한 Firmware update result log들을 database에 저장한다.
8. 여러 대의 Test System server로의 명령(Firmware update)네트워크로 전송하여 Firmware update를 실행한다.
9. Test System server로의 명령(test script start)네트워크로 전송하여 test를 실행한다.
10. Test System server에서 전송한 test script 실행 result log들을 database에 저장한다.
11. 여러 대의 Test System server로의 명령(test script 실행)네트워크로 전송하여 test를 실행한다.
12. 위의 내용들을 Web 요청에 따라 전송할 수 있는 서버를 만든다.

2.2.3 Web

1. Test 결과를 볼 수 있는 웹페이지의 목업을 만든다.(HTML, CSS)
2. Web 상에서 Test 결과를 Host server 에서의 data 를 받을 수 있게 페이지를 수정한다.
3. Firmware, Test script 별 결과를 볼 수 있는 웹페이지를 만든다.

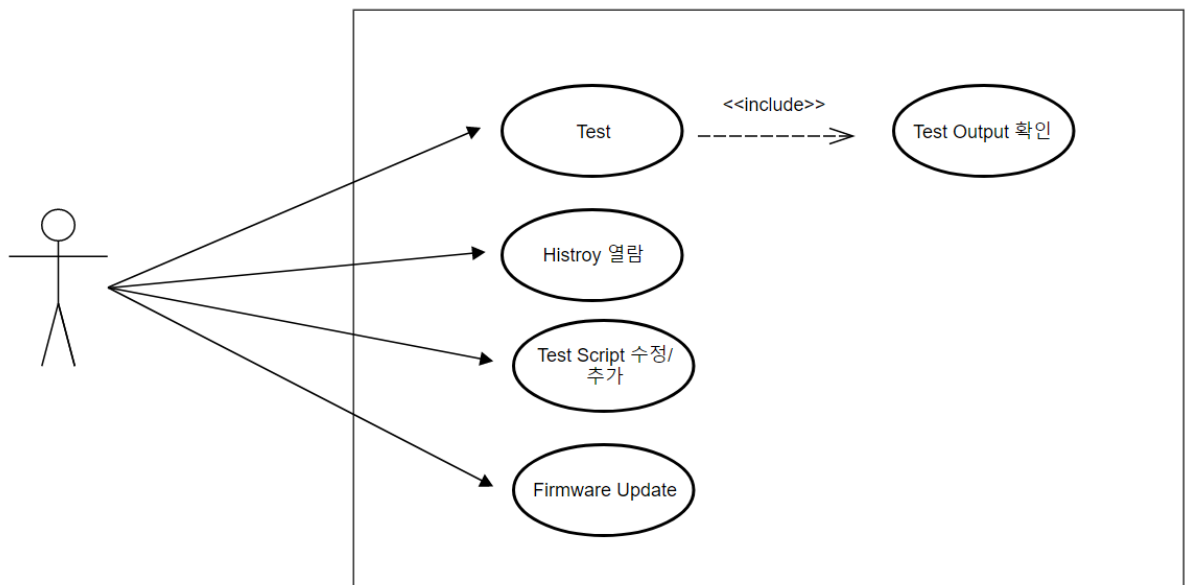
| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |


2.3 개발 결과

2.3.1 개발 결과 시나리오

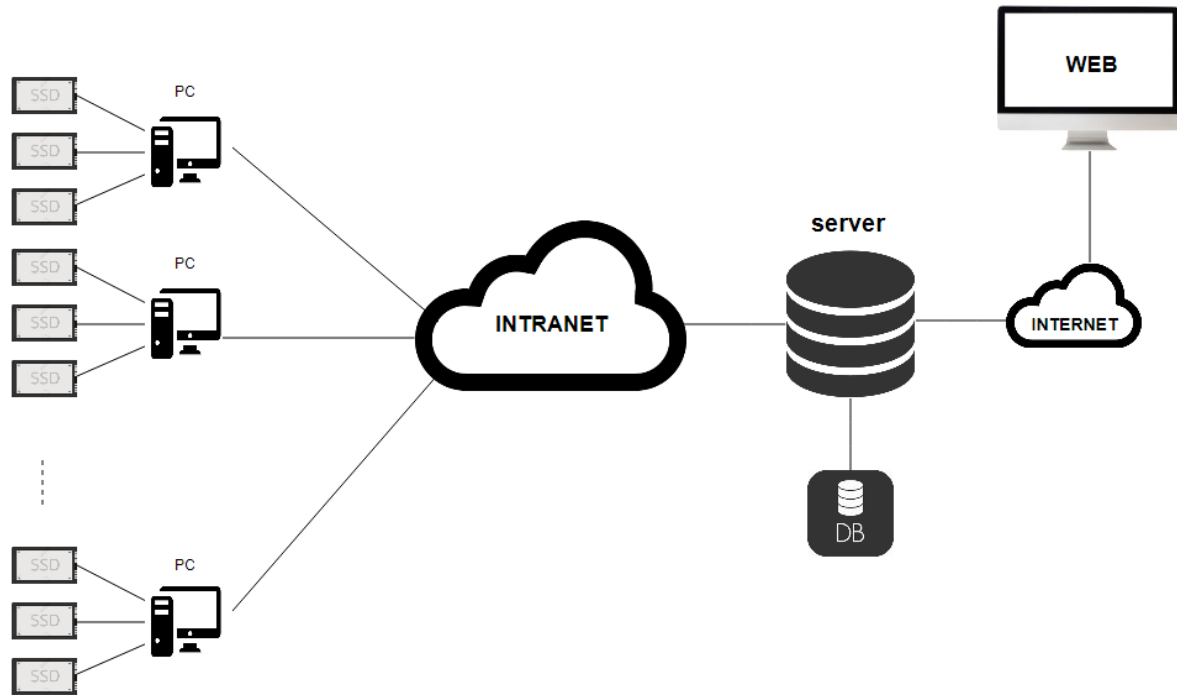
1. 관리자는 Test 를 실행한다.
2. 관리자는 Test 결과를 열람한다.
3. 관리자는 Test script 를 추가 및 변경한다.
4. 관리자는 과거 Test 결과를 열람한다.
5. 관리자는 firmware update 를 한다.

2.3.2 Use Case Diagram




| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |

2.3.3 시스템 설계도



2.4 기대효과 및 활용방안

현재 1:1 방식으로 진행되고 있는 SSD test이 remote 방식으로 변하게 되면 SSD test에 드는 시간과 인력이 현저하게 줄어들게 된다. 또한 Firmware, test script별 SSD test 결과를 요약해서 볼 수 있기 때문에 SSD test 을 통해 유저가 얻을 수 있는 정보가 간결해지고 분명해진다. 이는 SSD를 개발하고 있는 Fadu에서 더욱 효율적인 SSD 개발이 가능해지며, 또한 Firmware, test script의 version control 또한 수월해 질것으로 예상된다. Version control이 가능해지면 CI등의 프로세스 등도 가능할 것이다.

| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |

3 배경 기술

3.1 기술적 요구사항

3.1.1 프로젝트 개발환경

- 1) Client(SSD test) server
OS : Centos 6.9 server
Server : C기반 자체 개발 서버
 - 2) Host server
OS : Centos 6.7 server
Server : node.js(^v6.9.0 LTS)
Framework : express(^v4.0.0 LTS)
Database : Mongoddb
Proxy : Nginx
 - 3) Client
Browser : Chrome
Language : HTML + JavaScript(node.js)
- 프로젝트 버전관리
- 1) GitHub(public)
 - 2) Slack

3.2 현실적 제한 요소 및 그 해결 방안

3.2.1 소프트웨어

1. 속도
 - SSD Test를 할 때, 사용하는 리소스가 크기 때문에, Test 외의 performance에 영향이 최소화 되어야 한다. 따라서 performance에 영향이 최소화 되도록 가벼운 언어인 C언어 등으로 서버를 구현하여 Overhead를 최소화 하는 방안으로 작업한다.
2. 안정성
 - 동시에 여러 Test 장비를 가동했을 때, Host 서버가 부하에 견딜 수 있어야 한다. 또한 서비스 중 서버가 다운되면 로그에 대한 신뢰도가 떨어진다. 따라서 PM2 등 프로세스 매니저를 도입한다.

| | | | |
|---|-------------------------|------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 계획서 | | |
| | 프로젝트 명 | SSD Test 자동화 시스템 | |
| | 팀 명 | 1g's dormitory | |
| | Confidential Restricted | Version 1.0 | 2018-MAR-9 |

4 개발 일정 및 자원 관리

4.1 개발 일정

| 항목 | 세부내용 | 3 월 | 4 월 | 5 월 | 6 월 | 비고 |
|--------|-------------|-----|-----|-----|-----|----|
| 요구사항분석 | 요구 분석 | | | | | |
| | SRS 작성 | | | | | |
| 관련분야연구 | 주요 기술 연구 | | | | | |
| | 관련 시스템 분석 | | | | | |
| 설계 | 시스템 설계 | | | | | |
| 구현 | 코딩 및 모듈 테스트 | | | | | |
| 테스트 | 시스템 테스트 | | | | | |

4.2 일정 별 주요 산출물

| 마일스톤 | 개요 | 시작일 | 종료일 |
|-----------|--|------------|------------|
| 계획서 발표 | SSD Test PC 및 Host server PC 정하기 개발 환경 완성 (개발 PC OS 설치) 산출물 : 1. 프로젝트 수행 계획서 2. 프로젝트 기능 일람표 | ~ | 2018-03-08 |
| 설계 완료 | 시스템 설계 완료 산출물 : 1. 시스템 설계 사양서 | 2018-03-09 | 2018-03-23 |
| 1 차 중간 보고 | Test system server, remote firmware update 구현 완료 산출물 : 1. 프로젝트 1 차 중간 보고서 1 차분 구현 소스 코드 | 2018-03-24 | 2018-04-12 |
| 2 차 중간 보고 | Host server, web 구현 완료 산출물 : 1. 프로젝트 2 차 중간 보고서 2 차분 구현 소스 코드 | 2018-04-13 | 2018-05-18 |
| 테스트 | 시스템 통합 테스트 산출물: 시스템 테스트 결과 보고서 | 2018-05-18 | 2018-05-29 |
| 최종 보고서 | 최종 보고 | | 2018-06-01 |