



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	<i>SSD Test 자동화 시스템</i>
팀 명	<i>1g's dormitory</i>
문서 제목	계획서


Version	1.0
Date	2018-MAR-9

이름	권유상
----	-----

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


## 문서 정보 / 수정 내역

수정날짜	대표수정 자	Revision	추가/수정 항 목	내 용
2018-03-09	권유상	1.0	최초 작성	최초 작성

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

## 목 차

<b>1</b>	<b>개요</b>	<b>4</b>
1.1	프로젝트 개요	4
1.2	추진 배경 및 필요성	4
1.2.1	기존에 개발된 시스템 현황 및 한계점	4
1.2.2	개발할 시스템의 필요성	4
<b>2</b>	<b>개발 목표 및 내용</b>	<b>5</b>
2.1	목표	5
2.2	연구/개발 내용	5
2.2.1	Host Server & SSD Test Server	5
2.2.2	Database	5
2.2.3	Web	6
2.3	개발 결과	6
2.3.1	개발 결과 시나리오	6
2.3.2	Use Case Diagram	6
2.3.3	시스템 설계도	7
2.4	기대효과 및 활용방안	7
<b>3</b>	<b>배경 기술</b>	<b>8</b>
3.1	기술적 요구사항	8
3.2	현실적 제한 요소 및 그 해결 방안	8
<b>4</b>	<b>개발 일정 및 자원 관리</b>	<b>9</b>
4.1	개발 일정	9
4.2	일정 별 주요 산출물	9

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>캡스톤 디자인 I</b>	<b>계획서</b>		
	<b>프로젝트 명</b>	SSD Test 자동화 시스템	
	<b>팀 명</b>	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 소비도 적으며, 소음 또한 작기때문에 SSD시장이 더욱 활성화될 것으로 예상된다. 이에 따라서 여러 회사에서는 SSD 제품을 출시하고 있고, 현재 산학과제를 함께 진행하기로 한 회사인 Fadu 또한 SSD 제품을 개발하고 있기 때문에, 이 제품들을 검증하기 위한 시스템이 필수적이다. 현재 나와있는 Test Tool들은 대부분 local machine에서 테스트하기 때문에 대량으로 테스트하기에는 비효율적이다. 이러한 비효율성을 해결하기 위해, Fadu 에서는 이러한 기존 Tool들의 한계점을 보완한, 새로운 Test System을 산학과제로 다루기로 했다.

## 1.2 추진 배경 및 필요성


### 1.2.1 기존에 개발된 시스템 현황 및 한계점

현재는 SSD 테스트를 할 때, 하나의 Local Machine(PC)에서 SSD Benchmark Tool을 실행하기 때문에 비용(인력과 시간)이 크다. 또한 테스트 결과를 볼 수는 있지만 테스트 목적에 맞는 정보 또는 회사에서 필요한 정보들 외에도 많은 정보가 함께 있기 때문에 비효율적이다. 그리고 Log File들을 Dump file로 Database에 저장하기 때문에 Firmware/Test Script 별 SSD Test 이력관리가 어렵다.

### 1.2.2 개발할 시스템의 필요성

1. 한 PC당 하나씩 테스트를 진행하게 되면 수량이 많아 졌을 때 많은 시간을 소비하게 된다. 한 사람이 여러 대의 PC를 제어하여 한번에 다수의 SSD를 테스트 한다면 시간과 인력을 줄일 수 있다.
2. SSD Test 결과에 불필요한 정보까지 포함되어 있다. 테스트의 결과들 중 필요한 정보만 뽑아서 보여준다면 테스트 결과를 더 잘 활용할 수 있게 된다.
3. 테스트 이력을 저장해 놓으면 이전에 했던 테스트들과 비교/분석이 가능하다.
4. 테스트가 회사 내부에서만 가능하다면 시간 낭비가 크다. 테스트에 많은 시간이 소요되는데 그것을 회사 외부에서도 웹페이지를 통해 할 수 있다면 시간을 절약할 수 있다.

따라서 위와 같이 효율성을 높여, 더 많은 SSD를 빠르고 편리하게 Test해 줄 수 있는 System이 필요하다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>캡스톤 디자인 I</b>	<b>계획서</b>		
	<b>프로젝트 명</b>	SSD Test 자동화 시스템	
	<b>팀 명</b>	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. 원격으로 SSD Test가 실행되고 결과값을 원격으로 받아볼 수 있는 시스템
2. 동시에 여러 PC의 SSD Test를 수행하고 비동기식으로 결과값을 받아볼 수 있는 시스템
3. 원하는 테스트 결과 및 테스트 제어를 Web으로 할 수 있는 시스템
4. Firmware 버전관리와 Test Script의 관리가 테스트와 마찬가지로 원격으로 동시에 여러 SSD에 적용 가능한 시스템
5. 지금까지의 테스트 결과를 관리할 수 있는 시스템


### 2.2 연구/개발 내용

#### 2.2.1 Host Server & SSD Test Server

- Host Server는 중앙 서버로 각 컴퓨터와, Database, Web을 연결해 주는 역할이고, SSD Test Server (이하 Test Server) 는 각각의 컴퓨터에서 Host Server와의 통신을 담당하는 역할이다.
1. Local PC 내에서의 SSD Test Tool을 찾아서 이 Tool에 대한 I/O를 정리한다.
    - Local PC에서 Test Tool이 올바르게 실행되는지 확인하고 어떤 방식으로 테스트 하는지 확인하기 위함이다.
  2. SSD Test Tool의 데이터를 네트워크로 전송 가능한 message 형태로 가공한다
    - 어떤 데이터를 어떤 message 형식으로 가공할지 협의가 필요하다.
  3. SSD Test PC와 Host PC를 네트워크로 연결하고 기본적인 명령어(GET, POST 등)를 테스트 한다.
  4. 가공한 message를 송/수신이 가능한 서버를 개발한다.
    - SSD Test Server에서는 테스트 결과값을 송신하고 명령어를 수신한다.
    - Host Server에서는 테스트 결과값을 수신하고 명령어를 송신한다.
  5. 수신한 message로 원격으로 테스트를 실행할 수 있도록 Test Server를 개발한다.
  6. 현재까지의 내용을 1:n 상황에서도 작동할 수 있는 Host Server를 개발한다.
    - 1대의 local 컴퓨터에서 내린 명령이 n대의 컴퓨터에서 작동하도록 한다.

#### 2.2.2 Database

1. Host Server에서 수신 받은 message를 Log DB에 쌓는다.
  - 입출력 형식을 정한 후 스키마와 테이블을 정의한다.
2. Script 수정 및 추가를 위한 Script repository와 Remote Firmware Update를 위한 Firmware repository를 생성한다
3. Script repository에서 SSD Test PC로 Script를 전송하고 전송된 Script로 테스트할 수

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

있도록 Test Server를 수정한다.

- 이전 버전에서는 Test PC에 존재하는 Script를 실행하지만, 이번 버전부터 Host PC에서 Script를 전송하여 수신 받은 Script를 실행하는 것을 목표로 한다.
- 4. Firmware를 원격에서 SSD Test PC로 전송하고 Firmware Update가 작동하게 Test Server와 Host Server를 수정한다.

### 2.2.3 Web

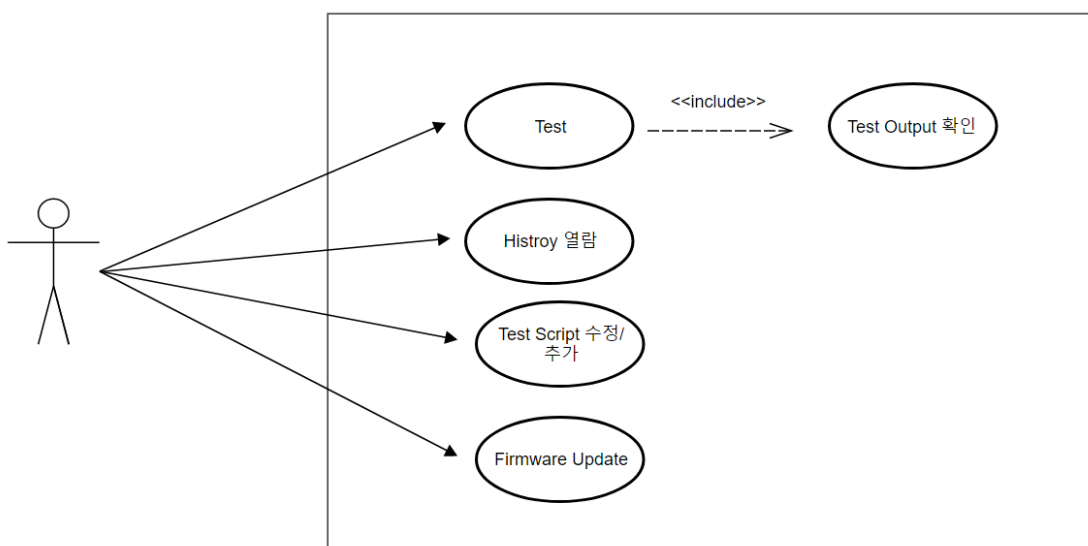
1. 테스트 결과를 볼 수 있는 웹페이지를 만든다.
  - HTML+CSS로 제작예정
2. 웹페이지와 Host Server가 통신할 수 있도록 만든다.
3. 테스트 결과를 웹상으로 보여줄 수 있게 하는 기능을 Host서버에 추가한다.

## 2.3 개발 결과

### 2.3.1 개발 결과 시나리오

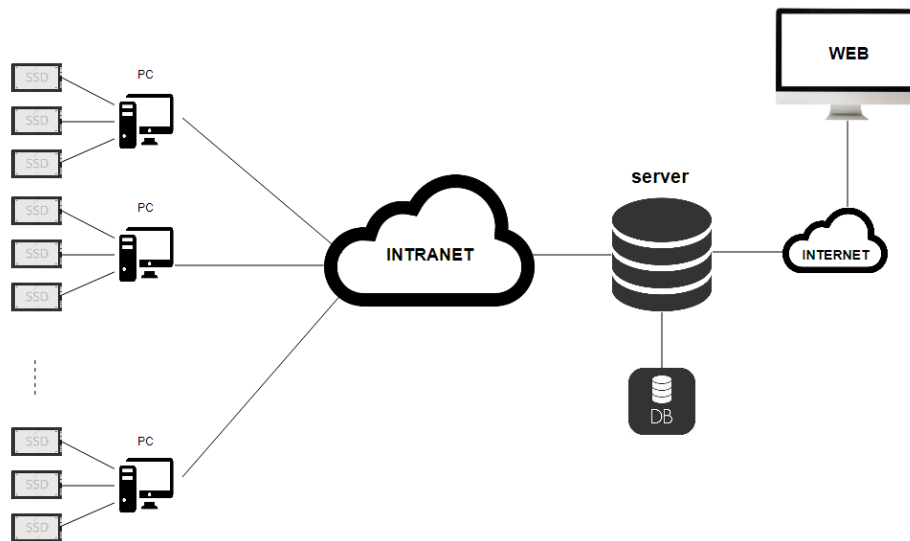
1. 관리자는 Test 를 실행한다.
2. 관리자는 Test 결과를 열람한다.
3. 관리자는 Test Script 를 추가 및 삭제, 변경한다.
4. 관리자는 과거 Test 결과를 열람한다.
5. 관리자는 Firmware Update 를 한다.

### 2.3.2 Use Case Diagram



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

### 2.3.3 시스템 설계도



## 2.4 기대효과 및 활용방안

- 여러 개의 SSD를 한 컴퓨터로 테스트 할 수 있게 된다.
  - 한 대의 local 컴퓨터로 다른 여러 컴퓨터들의 테스트를 컨트롤 할 수 있게 하여, 하나씩 테스트하는 것보다 편리하게 된다.
- 회사 내부가 아닌 다른 곳에서도 SSD 테스트가 가능하게 된다.
  - 회사 내부에서만 가능하던 테스트가 인터넷이 되는 어느 곳이든 가능해져서 효율성이 높아 진다.
- 테스트뿐만 아니라 Firmware Update도 테스트 방식과 같게 작동하게 된다.
- 과거 테스트 결과들과 필요한 테스트 결과를 볼 수 있게 된다.
  - 여러 테스트 결과 중 회사에서 필요한 정보만 볼 수 있게 되고, 과거 테스트 결과를 저장해 놓고 필요할 때 언제든지 볼 수 있게 된다.

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

### 3 배경 기술

#### 3.1 기술적 요구사항

##### - 프로젝트 개발환경

- 1) 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 : MongoDB  
Proxy : Nginx
- 3) Client(Web)  
Browser : Chrome  
Language : HTML + JavaScript(node.js)

##### - 프로젝트 버전관리

- 1) GitHub(public)
- 2) Slack

#### 3.2 현실적 제한 요소 및 그 해결 방안

##### 소프트웨어

###### 1. 속도

- SSD Test를 할 때, 사용하는 리소스가 크기 때문에, 테스트 외의 Performance에 영향이 최소화 되어야 한다. 따라서 Performance에 영향이 최소화 되도록 가벼운 언어로 Test Server를 구현하여 Overhead를 최소화 하는 방안으로 작업한다.

###### 2. 안정성

- 동시에 여러 테스트 장비를 가동했을 때, 그 메시지들을 처리하는 서버 즉, Host Server가 부하에 견딜 수 있어야 한다. 또한 서비스 중 서버가 다운되면 로그에 대한 신뢰도가 떨어진다. 따라서 PM2등 프로세스 매니저를 도입한다.



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>캡스톤 디자인 I</b>	<b>계획서</b>		
	<b>프로젝트 명</b>	SSD Test 자동화 시스템	
	<b>팀 명</b>	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) <b>산출물 :</b> 1. 프로젝트 수행 계획서 2. 프로젝트 기능 일람표	~	2018-03-08
설계 완료	시스템 설계 완료 <b>산출물 :</b> 1. 시스템 설계 사양서	2018-03-09	2018-03-23
1 차 중간 보고	개발/연구내용 1~6 구현 완료 <b>산출물 :</b> 1. 프로젝트 1 차 중간 보고서 1 차분 구현 소스 코드	2018-03-24	2018-04-12
2 차 중간 보고	개발/연구내용 6~14 구현 완료 <b>산출물 :</b> 1. 프로젝트 2 차 중간 보고서 2 차분 구현 소스 코드	2018-04-13	2018-05-18
테스트	시스템 통합 테스트 <b>산출물:</b> 시스템 테스트 결과 보고서	2018-05-18	2018-05-29
최종 보고서	최종 보고		2018-06-01