



국민대학교  
전자정보통신대학  
컴퓨터공학부

#### CONFIDENTIALITY/SECURITY WARNING


이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 캡스톤 디자인Ⅰ 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	1.1
Date	2018-04-12

이름	김주용
----	-----

 <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	김주용	1.0	최초 작성	최초 작성
2018-04-12	김주용	1.1	수정	피드백 후 재 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
<b>2</b>	<b>개발 목표 및 내용</b>	<b>5</b>
2.1	목표	5
2.2	연구/개발 내용	5
2.3	개발 결과	6
2.3.1	개발 결과 시나리오	6
2.3.2	시스템 설계도	6
2.4	기대효과 및 활용방안	6
<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 우리 주변에서도 여러 매체 광고등을 통해 많이 대두되고 있는 SSD의 중요성은 HDD 시장은 곧 침체 될 것 이라는 사실을 떠올리게 한다. SSD 산업이 발전함에 따라 SSD를 만드는 회사가 늘어났고 자연스레 SSD TEST TOOL(SSD의 성능 검증)에 대한 수요도 함께 증가하게 되었다. 물론 기존에 출시되어 있는 SANBLAZE, OAKGATE등 여러 benchmarking TEST 장비들이 있지만, 이러한 장비들의 장점을 최대한 종합하고 불필요한 작업들을 줄이기 위해 FADU 에서 CAPSTONE 을 통해 새로운 SSD TEST 자동화 시스템을 구축하기로 하였다.

## 1.2 추진 배경 및 필요성

### 1.2.1 기존에 개발된 시스템 현황 및 한계점

현재 회사에서는 보통 SSD의 성능을 TEST하는 회사 내 개발 TOOL과 상용화 되어있는 장비를 사용하고 있다. 회사 내부망을 통해서 SSD들이 붙어있는 측정장비에 접속하여 TEST를 사용하고 있고 TEST 후 결과를 LOG 파일 형식으로 저장 가능하고 결과 DATA를 PASS/FAIL로 확인할 수 있다.

여기에서 찾아 볼 수 있는 한계점은

첫째, 회사 외부에서 원격으로 TEST 하는 WEB UI 가 없다.

둘째, 여러 장비를 통한 SSD TEST 결과를 한꺼번에 종합할 수 없고 Client pc와 test 장비간 1:1 방식 사용으로 인력과 시간의 낭비가 크다

셋째, TEST 결과 DATA에서 회사측에서 필요한 DATA만 보여지지 않는다.

### 1.2.2 개발할 시스템의 필요성


위에서 설명한 한계점들을 토대로 보면

첫째, 회사 내부망에 한정 되어 있는 접속망을 인터넷 통신을 통해 사용자가 웹페이지를 통해 접근하여 회사 외부에서도 TEST를 진행 할 수 있도록 한다.

둘째, 여러 장비를 TEST한 결과들을 가공하여 한번에 web ui tool 에서 Control 할 수 있도록 한다.

셋째, TEST 결과를 VISUALIZING 하는 방식을 좀더 보기 쉽게 하고 (그래프화) , 많이 필요로 하는 정보를 우선으로 보여줄 수 있도록 한다.

이렇게 크게 3가지 측면으로 SSD TEST 시스템 의 필요성을 정리 할 수 있다.

 <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

기존의 TEST TOOL 시스템의 장점을 종합하고 단점은 피하는 방식으로 새로운 시스템 TOOL의 BASE로 하고 Requirement를 통해 기능을 더하여 SSD 산업에서 경쟁력을 가지는 TOOL을 개발한다. 이를 위해서 크게 4가지로 기능적 측면을 볼 수 있다

1. Remote Test :
  - Test를 종합하는 역할의 Host Server를 구현하고 회사 외부에서 인터넷을 통해 접근을 가능하게 하여 어디서든 원격으로 Test를 진행 할 수 있다.
2. Multiple Test:
  - Test pc 의 SSD TEST서버들을 하나의 Host Server로 연결하여 다대일 방식의 Test가 가능하도록 한다.(하나의 중앙 server와 여러 대의 client SSD TEST PC)
3. Test Data Visualizing:
  - Test 결과를 보다 효율적(필요도가 높은 정보 우선)으로 정리하여 보여 줄 수 있도록 한다(그래프).
4. FW버전 관리:
  - 하나의 Host Server에서 연결된 여러 SSD Test pc의 Firmware update를 일괄적으로 한번에 진행한다.


### 2.2 연구/개발 내용

#### Client -Server 통신 구축

1. SSD TEST PC에서 TEST 결과 Raw Data를 저장한다.
2. 저장된 Raw Data를 네트워크로 전송 가능한 특정 message 형태(text나 script로) 가공 해 준다.
3. SSD test PC와 구현할 Host server를 사내 Network을 통하여(ethernet) 연결한다.
4. Host Server에서 test command message를 전송하여 test를 진행하고 Raw Data값을 message 형식으로 전달받아 저장한다.
5. 저장된 데이터를 Host Server 내에서 다시 가공하여 외부 네트워크(인터넷)를 통해 제작할 Web tool로 보여준다.
6. 현재까지의 내용을 1(Host Server): n (Client Test PC) 상황에서도 작동할 수 있는 Host서버를 개발한다.
7. Host Server에서 수신 받은 message를 Log DB에 쌓는다.
  - I/O가 정리되고 난 후 스키마와 테이블을 정의한다.

#### Web UI Tool

8. Web Tool 내에서 직접 Test script 수정 및 추가 작업이 가능하도록 host 서버를 구축 해 준다.
9. Web Tool 내에서 연결된 n 대의 Test PC 들을 일괄적으로 관리 할 수 있도록 host 서버 내에서 data를 가공해 주도록 한다
10. 가공된 Data를 사용자가 필요한 정보를 우선으로 전달하고 그래프나 flow chart를 통한 visualizing 기법을 사용하여 보여준다.

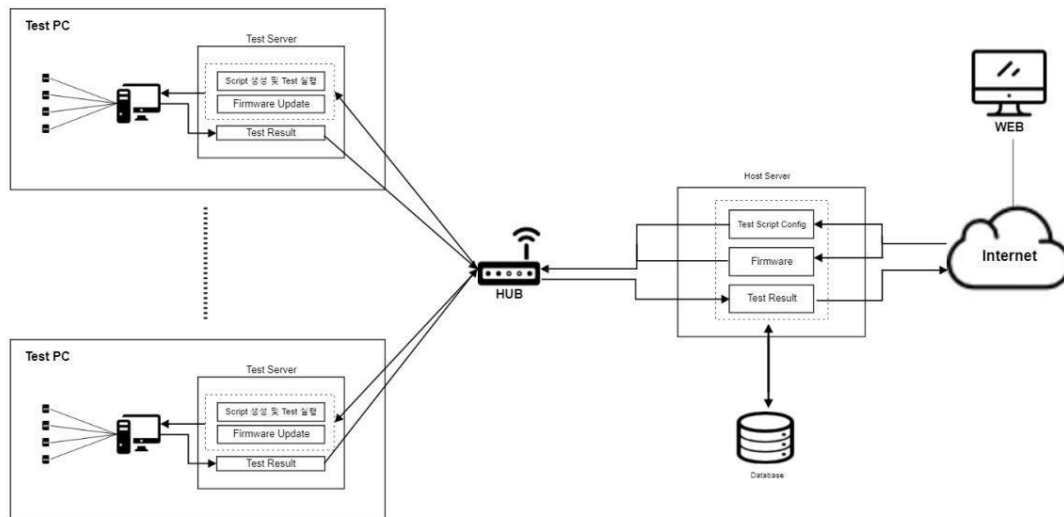
 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>캡스톤 디자인 I</b>	<b>계획서</b>		
	<b>프로젝트 명</b>	SSD Test 자동화 시스템	
	<b>팀 명</b>	1g's dormitory	
	Confidential Restricted	Version 1.0	2018-MAR-9

## 2.3 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 시스템 설계도



## 2.4 기대효과 및 활용방안

- 기존의 회사에서 사용하는 방식은 Client SSD Test 장비 한대 당 하나의 관리PC를 두어 1:1 방식으로 SSD TEST를 진행 하였기 때문에 여러 TEST 장비들의 결과를 각각 따로 확인하는 방식이다. 이는 인력과 시간에 많은 제약 조건을 주게 된다.
- 또한, 회사 외부에서 인터넷을 통한 관리 서버가 따로 존재하지 않아 TEST를 진행할 시 관리자가 사내에 상주하고 있어야 한다는 단점이 있다. 따라서 이러한 단점들을 분명히 보완해 줄 수 있도록 새롭게 구축한 시스템이 SSD TEST 자동화 시스템이다.

### 1) 기대효과

 <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

- 다량의 SSD TEST를 진행 할 때 각각 결과를 확인 할 필요가 없이 통합서버에 연결되어 있는 WEB UI TOOL 화면을 통해 한번에 결과를 종합하여 확인 할 수 있다.
- 통합서버에 인터넷으로 연결된 WEB UI TOOL으로 회사 외부에서도 언제든지 TEST를 진행 할 수 있다.
  - N:1 방식의 TEST진행 서버 구축으로 FIRMWARE 업데이트 시에 일괄적으로 한번에 업데이트가 가능하다.

## 2) 활용방안

- 다수의 SSD TEST를 진행하면 한번에 그 결과를 종합하여 비교 할 수 있으므로 제품별 성능 비교가 용이하다.
- 외부 출장 기간 동안에도 관리자가 웹을 통해 어디서든 TEST를 실행 할 수 있다.
- 펌웨어 업데이트를 한번에 함으로써 사용자가 번거롭지 않다.
  - 여러 SSD TEST를 종합하여 결과값을 그래프로 보여줌으로써 사용자가 한 눈에 알아보기 쉽다.

 <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 CLIENT

OS : Centos 6.9

TEST-CLIENT : golang

2) Host server

OS : Centos 6.7

TEST-SERVER : golang

WEB-Server : node.js(^v6.9.0 LTS)

Framework : martini

Database : Mongodb/mysql

Proxy : Nginx

3) Client

WEB-CLIENT : WEB BROWSER

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, 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, 그 메시지들을 처리하는 서버가 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