



[2015SMA\_T3]

CTIP

201011320 김용현  
201111360 손준익  
201111347 김태호

# INDEX.

---

---

1. CTIP

---

---

2. JUnit

---

---

3. Hudson

---

---

4. Mantis

---

---

5. Git, Github, Source Tree.

---

---

6. Q&A

---

---

---

# 1

## CTIP

Introduce of CTIP

---

### CTIP 이란?

**Continuous Test & Integration Platform**

지속적인 테스트와 그에 따른 통합적인 개발환경을 제공하는 Platform

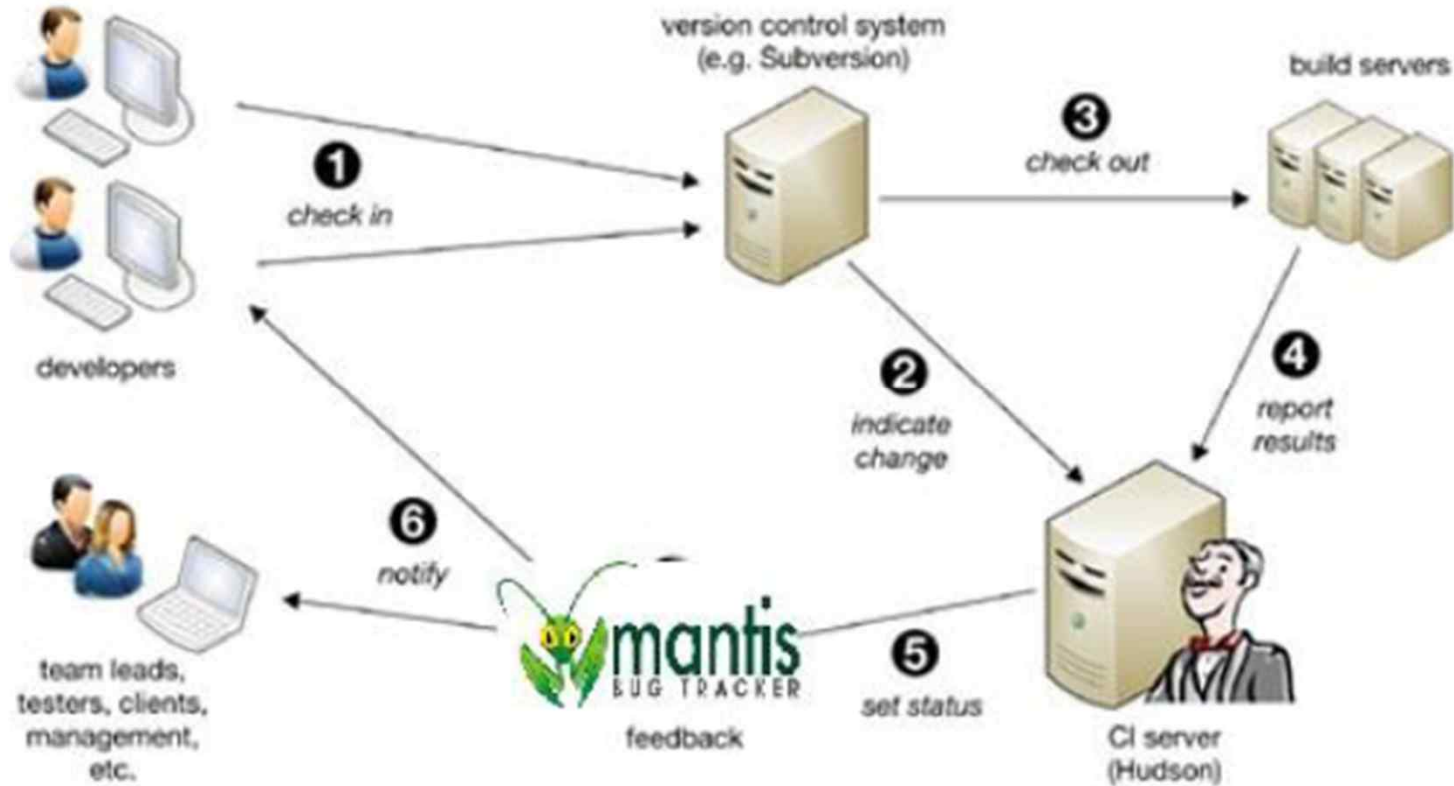
CI서버를 통한 지속적인 통합환경과 Build Automation을 기반으로 한다.

품질도구들을 통하여 코드의 품질을 검토 할 수 있다.

Build 결과와 Testing Output을 해당 프로젝트 관련자들에게 배포한다.

# 1 CTIP

Introduce of CTIP



---

# 1

## CTIP

Introduce of CTIP

---

### CTIP 구성 요소

CI 서버

코드 품질 관리

소스 코드 버전관리(SVN, CVS, Git)

빌드 및 배포

---

# 1

## CTIP

Introduce of CTIP

---

| Category                 | Tool                              |
|--------------------------|-----------------------------------|
| CI Server                | Hudson                            |
| Unit Testing             | Junit                             |
| Build                    | Hudson                            |
| Version Control          | Git                               |
| Bug Tracking & Community | Mantis                            |
| Static Analysis          | Eclipse tptp, Sonar, Cppcheclipse |

---

# 1

## CTIP

Introduce of CTIP

---

### 성공적인 CI 수행 조건

**Source Repository(단일 소스 저장소) 유지**

**Build Automation**

**모든 사용자는 매일 작업 내용을 Commit**

**모든 Commit은 통합 서버(CI) 메인 라인에 반영**

**각 Build는 빠르게 수행되어야 한다.**

**운영 환경과 비슷한 환경에서 테스트**

**최신 결과물에 쉽게 접근 할 수 있어야 한다.**

**현재 Build 상황을 쉽게 알 수 있어야 한다.**

---

# 1

## CTIP

Introduce of CTIP

---

### CTIP's Advantage

**위험을 줄일 수 있다.**

**수동으로 수행해야 하는 반복 작업을 줄일 수 있다.**

**시간과 장소에 구애 받지 않고 배포 할 수 있는 소프트웨어를 만들 수 있다.**

**프로젝트에 대한 더 나은 가시성을 제공해 준다.**

**코드 품질에 대한 더 높은 신뢰성을 제공해 준다.**



---

# 2 JUnit

JUnit ?

---

## JUnit 이란?

**Java에서 사용하는 단위 테스트(Unit test)를 위한 Frame Work**

**단위 모듈이 정확히 구현되었는지 확인가능**

**2.1 버전 이후부터는 Eclipse안에 내장.**

---

# 2 JUnit

JUnit ?

---

## JUnit 의 특징

|            |              |
|------------|--------------|
| 주요 기능      | 지원 내용        |
| 테스트 범위     | 단위 테스트       |
| TDD 환경 지원  | 지원           |
| 코드 지원      | 지원           |
| Test Suite | 지원           |
| UI Report  | 지원           |
| Code Trace | 지원, 실시간 추적가능 |

---

# 2 JUnit

JUnit Test Method

---

## JUnit Test Method

|                                     |                                                                                                                                     |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| <code>assertEquals(a,b)</code>      | 객체 a,b가 서로 일치하는지 확인                                                                                                                 |
| <code>assertArrayEquals(a,b)</code> | 배열 a,b가 일치하는지 확인(순서 포함)                                                                                                             |
| <code>assertNull(a)</code>          | 객체가 Null을 참조하는지 시험                                                                                                                  |
| <code>assertNotNull(a)</code>       | 객체가 실존하는 객체를 확인하는지 시험                                                                                                               |
| <code>assertTrue(a)</code>          | 조건식 a가 참인지 확인                                                                                                                       |
| <code>assertFalse(a)</code>         | 조건식 a가 False인지 확인                                                                                                                   |
| <code>assertSame(a,b)</code>        | a가 참조하는 객체를 b도 참조하는지 시험                                                                                                             |
| <code>assertNotSame(a,b)</code>     | a와 b가 서로 다른 객체를 참조하는지 시험                                                                                                            |
| <code>Fail()</code>                 | 테스트 결과를 실패로 처리                                                                                                                      |
| 기타                                  | <a href="http://junit.sourceforge.net/javadoc/org/junit/Assert.html">http://junit.sourceforge.net/javadoc/org/junit/Assert.html</a> |

---

# 2 JUnit

J Unit of Annotation

---

## J Unit of Annotation

| Annotation                     | 설명                                                                     |
|--------------------------------|------------------------------------------------------------------------|
| @Test                          | Unit Test를 수행하는 대상 method                                              |
| @Before                        | 각 Unit test의 method 실행 전에 실행되는 method                                  |
| @After                         | 각 Unit test의 실행 후에 실행되는 method                                         |
| @BeforeClass                   | Class안에 정의된 모든 method에 대해서 Test 전, 후에 한번만 호출된다. 객체 생성 등에 사용.           |
| @AfterClass                    |                                                                        |
| @Ignore                        | 테스트를 수행하지 않을 method                                                    |
| @RunWith(value=class)          | Unit Test 클래스를 실행하기 위한 러너(Runner)를 명시적으로 지정할 수 있다.                     |
| @SuiteClasses<br>(value=class) | 보통 여러 개의 Test Class를 수행하기 위해 쓰인다. @RunWith를 이용해 Suite Class를 러너로 사용한다. |
| @Parameter                     | 하나의 method에 대해 다양한 테스트 값을 한꺼번에 실행시키고자 할 때 사용한다.                        |

# 2 J Unit

사용 예제

```
@BeforeClass // 테스트 클래스 시작전에 실행되는 method
public static void Setup() throws Exception{
    textReader = new Testcode();
    System.out.println("@BeforeClass Annotation : Testcode 객체 생성");

    textReader.connect(); //연결
    System.out.println("connection 연결");
}
```

```
@AfterClass
public static void Setoff() throws Exception{
    textReader.disconnect(); //연결해제
    System.out.println("@AfterClass Annotation : 연결 해제");
}
```

```
@Before
public void Setup_path(){
    text_path="test.txt";
    text_path2="nontext.txt";
    System.out.println("@Before Annotation : text path 설정");
}
```

```
@After
public void Setoff_path(){
    text_path=null;
    text_path2=null;
    System.out.println("@After Annotation : text path,2 초기화");
}
```

```
@BeforeClass Annotation : Testcode 객체 생성
connection 연결
@Before Annotation : text path 설정
@After Annotation : text path,2 초기화
@Before Annotation : text path 설정
@After Annotation : text path,2 초기화
@Before Annotation : text path 설정
@After Annotation : text path,2 초기화
@Before Annotation : text path 설정
@After Annotation : text path,2 초기화
@Before Annotation : text path 설정
@After Annotation : text path,2 초기화
@AfterClass Annotation : 연결 해제
```

---

# 3 Hudson

hudson

---

## Hudson

**Build 환경 Tool.**

**Continuous Integration Tool : 정기적으로 혹은 특정시기에 자동으로 Build.**

**Git과 Source Tree로 버전관리**

# 3 Hudson

hudson

## Build 방법

### 계정 권한 설정

Admin으로 사용할 계정 생성 후,  
Manage Hudson – configure  
security 에서 추가



## Configure Security

Enable security

TCP port for JNLP slave agents  Fixed :   Random  Disable

Markup Formatter

Raw HTML

Treat the text as HTML and use it as is without any translation

Access Control

### Security Realm

- Hudson's own user database
  - Allow users to sign up
  - Notify user of Hudson account creation
- LDAP
- Delegate to servlet container

### Authorization

- Logged-in users can do anything
- Job-based Matrix Authorization Strategy
- Anyone can do anything
- Matrix-based security
- Team based Authorization Strategy

Team based strategy expects at least one System Administrator who will have full privilege then a group can be added and all members of the group will have full privilege.

**Current System Administrators:**

1. gabriel0218

---

# 3 Hudson

hudson

---

## Build 방법

계정 권한 설정

Manage team 에서 작업할  
사용자를 추가.

The screenshot shows the Hudson Manage Teams interface. At the top, there are tabs for 'Manage Teams', 'Manage Jobs', 'Manage Views', and 'Manage Nodes'. Below these is an 'Add New Team' button. A team named 'team' is highlighted in orange. Below the team name, there is a 'Delete Team' button. Underneath, there are tabs for 'Members', 'Jobs', 'Views', and 'Nodes'. An 'Add New Member' button is visible. A table lists the members and their permissions:

| User        | Admin | Job Permissions |        |           |       | View Permissions |        |           | Node Permissions |        | Actions |           |
|-------------|-------|-----------------|--------|-----------|-------|------------------|--------|-----------|------------------|--------|---------|-----------|
|             |       | Create          | Delete | Configure | Build | Create           | Delete | Configure | Create           | Delete |         | Configure |
| gabriel0218 |       | ✓               | ✓      | ✓         | ✓     |                  |        |           |                  |        |         |           |



---

# 3 Hudson

hudson

---

## Build 방법

### 새 작업 만들기

### 새 작업을 만들고 작업을 수행할 팀을 지정

여기(영문)을 보세요.' 4. '기존 작업 복사': 'Copy from' followed by an empty input field. At the bottom is an 'OK' button."/>

**New Job**

작업명

Team name

- Build a free-style software job**  
이것은 Hudson의 주요 기능입니다. Hudson은 어느 빌드 시스템과 어떤 SCM(항상관리)으로 묻힌 당신의 프로젝트들을 빌드할 것이고, 소프트웨어 빌드보다 더 좋습니다.
- Build multi-configuration job**  
다양한 환경에서의 테스트, 공개용 특성 빌드, 기타 응용 처럼 다수의 서로다른 환경환경이 필요한 프로젝트에 적당함.
- Monitor an external job**  
이 유형의 작업은 원격 장비처럼 Hudson 외부에서 동작하는 프로세스의 실행을 기록하는 것을 허용합니다. 그렇게 설정되어서, 기존의 자물 시스템의 대서보드 습니다. 자세한 설명은 [여기\(영문\)](#)을 보세요.
- 기존 작업 복사**  
Copy from

---

# 3 Hudson

hudson

---

## Build 방법

계정 권한 설정

작업 설정에서 코드를 저장할 git repository를 설정

Source Code Management

None

Git

Repositories

URL of repository

Add

---

# 3 Hudson

hudson

---

## Build 방법

### 작업 설정

**Build Triggers – Schedule**란  
에 분, 시, 일, 월, 요일 순서로  
자동 Build할 시간을 기록

\*는 전부를 뜻함.

**Build Triggers**

- Build after other jobs are built
- Trigger builds remotely (e.g., from scripts)
- Build periodically

Schedule

|    |                                     |
|----|-------------------------------------|
| 분  | 0 ~ 59 사이의 값을 입력합니다.                |
| 시  | 0 ~ 23 사이의 값을 입력합니다.                |
| 일  | 1 ~ 31 사이의 값을 입력합니다.                |
| 월  | 1 ~ 12 사이의 값을 입력합니다.                |
| 요일 | 0 ~ 7 사이의 값을 입력합니다. (0과 7은 일요일입니다.) |

# 3 Hudson

hudson

## Build 방법

Build

Build now를 클릭하면 즉시 Build 시작.

Back to Main Dashboard

- Status
- Changes
- Workspace
- Build Now
- Delete Job
- Configure

Build History (12/12)

| #   | Time                    | Status  |
|-----|-------------------------|---------|
| #12 | 2015. 3. 18 오전 1:28:52  | Success |
| #11 | 2015. 3. 18 오전 1:25:07  | Success |
| #10 | 2015. 3. 18 오전 1:18:20  | Success |
| #9  | 2015. 3. 18 오전 1:12:28  | Failure |
| #8  | 2015. 3. 18 오전 1:10:44  | Failure |
| #7  | 2015. 3. 18 오전 1:10:07  | Failure |
| #6  | 2015. 3. 18 오전 12:57:16 | Failure |
| #5  | 2015. 3. 18 오전 12:27:56 | Failure |
| #4  | 2015. 3. 17 오후 11:41:55 | Failure |
| #3  | 2015. 3. 17 오후 11:40:32 | Failure |
| #2  | 2015. 3. 17 오후 10:29:28 | Failure |
| #1  | 2015. 3. 17 오후 10:28:46 | Failure |

for all for failures



Back to Main Dashboard

- Status
- Changes
- Workspace
- Build Now
- Delete Job
- Configure

Build History (12/12)

| #   | Time                    | Status  |
|-----|-------------------------|---------|
| #12 | 2015. 3. 18 오전 1:28:52  | Success |
| #11 | 2015. 3. 18 오전 1:25:07  | Success |
| #10 | 2015. 3. 18 오전 1:18:20  | Success |
| #9  | 2015. 3. 18 오전 1:10:29  | Failure |
| #8  | 2015. 3. 18 오전 1:09:44  | Failure |
| #7  | 2015. 3. 18 오전 1:09:07  | Failure |
| #6  | 2015. 3. 18 오전 12:57:16 | Failure |
| #5  | 2015. 3. 18 오전 12:27:56 | Failure |
| #4  | 2015. 3. 17 오후 11:41:55 | Failure |
| #3  | 2015. 3. 17 오후 11:40:32 | Failure |
| #2  | 2015. 3. 17 오후 10:29:28 | Failure |
| #1  | 2015. 3. 17 오후 10:28:46 | Failure |

for all for failures

Job team.testjob

- Workspace
- Revert Changes
- Latest Console output

Permalinks

- Last build (#12), 35 sec ago
- Last stable build (#12), 35 sec ago
- Last successful build (#12), 35 sec ago
- Last failed build (#9), 31 min ago
- Last unsuccessful build (#9), 31 min ago

---

# 3 Hudson

hudson


---

## Build 방법


### Build 결과


Build 시간, 코드, 변경사항, 에러 등 확인가능.



 빌드 #12 (2015. 3. 18 오전 1:38:52)

 No changes.

 사용자 박호모에 의해 시작됨

 **Revision:** 427df1625cf98b38aa2cbe68b5bddcb36586b5e1  
• origin/master

---

# 4 Mantis

Mantis

---

## Mantis 란?

**MantisBT는 인기있는, 웹 기반 오픈소스 Bug tracking system.**

**PHP 기반으로 구축.**

**서버 측에서 Linux, Windows, Mac OS X 지원**

**Chrome, Firefox, Safari, Opera, IE7+ 와 호환**

**GNU General Public License(GPL)을 따름**

---

# 4 Mantis

Mantis

---

## Bug tracking System 이란?

**결함이 발견된 때부터 해결된 때까지의 과정을 기록하고 추적**

**결함추적(Defect tracking)이라고도 부름.**

**개별적 수준, 즉 각각의 결함을 추적하며, 동시에 공개된 결함의 개수, 해결된 비율, 결함을 하나 해결하는데 소요되는 평균 시간과 같은 통계적 수준에서도 이루어짐.**

---

# 4 Mantis

Mantis

---

## Bug tracking System 필요성

Web을 통해 접근이 용이하고 쉽게 사용할 수 있음.

모든 버그는 DB에 보관되며 추적가능.

버그에 수많은 정보와 파일 첨부, 해당 버그의 담당자가 누구이며 현재 진행 업무 파악 가능

버그가 줄어드는 것을 통계로 확인 -> 버그의 통계자료를 문서 작성에 활용

외국에서도 원격으로 접속해서 버그를 보고 가능프로젝트 관리자가 각 개발자 별 업무를 조율해 줄 수 있는 용도로 활용



---

# 4 Mantis

Mantis

---

## Issue tracking System ?

**필요에 따라 이슈들의 목록을 관리하고 유지하는 컴퓨터 소프트웨어 패키지**

**일반적으로, 보고된 고객의 이슈들이나 조직의 다른 직원들이 보고한 이슈를 생성하거나 업데이트, 이슈를 해결하기 위해 사용**

**Bug Tracking System과 유사**

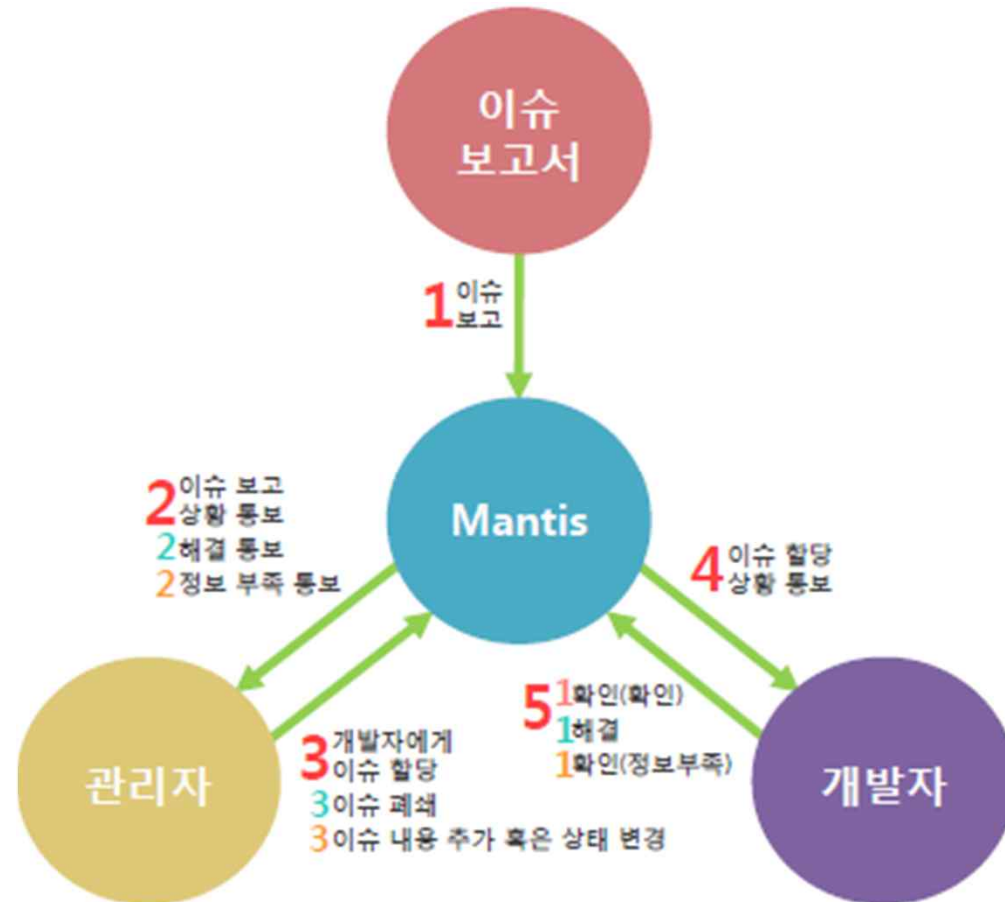
**Mantis는 Bug Tracking 중심이어서, 이슈관리가 어렵고 다른 툴과의 연동이 어려움 -> Redmine 추천.**

# 4 Mantis

Mantis

## Issus Tracking System

### 운영 다이어그램



---

# 5 Git, Source Tree/ Git hub

Git, Source Tree, GitHub

---

## Git, Git Hub, Source Tree?

**Git** : Version 관리 툴. Repository를 생성 관리 가능.

**Git Hub** : Web상에서 Version에 관한 정보를 생성 관리 가능.

**Source Tree** : 이미 만들어진 Git 저장소와 연동하여 Version관리 하는 툴. 편리한 Interface를 제공하며 Comment 할 수 있다.

# 5 Git, Github, Source Tree

Git, Github, Source Tree

ryums0227 / 2014\_pp

Unwatch 2

21 commits

1 branch

0 releases

1 contributor

branch: master 2014\_pp / +

확인용 out.print 삭제

aksjm2 authored on Jun 15 2014

latest commit 7a16f5607e

|                           |                    |               |
|---------------------------|--------------------|---------------|
| .settings                 | 버전업~               | 11 months ago |
| WebContent                | 확인용 out.print 삭제   | 11 months ago |
| build/classes/information | 마일리지정보랑, OnOff 수정함 | 11 months ago |
| src/information           | 마일리지정보랑, OnOff 수정함 | 11 months ago |
| .classpath                | 버전업~               | 11 months ago |
| .project                  | commit 1           | 11 months ago |

Help people interested in this repository understand your project by adding a README!


Add a README

---

# 6 질의응답

Question & Answer

---



Question  
& Answer



**감사합니다.**

[2015SMA\_T3]

---

---