QE-BOK Quality Engineering-Body Of Knowledge

For external Education

Continuous Integration for Staged Build (for Java)

Productivity Innovation Lab,
NHN Corp
2011.12

1. Course Introduction

1.1 Objective

- Understand NHN Quality Criteria
- Understand CI / Staged Build
- Learn how to set up Hudon(Jenkins) and mandatory plugins

1.2 WHO AM I

❖ **JunHo Yoon**

❖ **Work Experience**

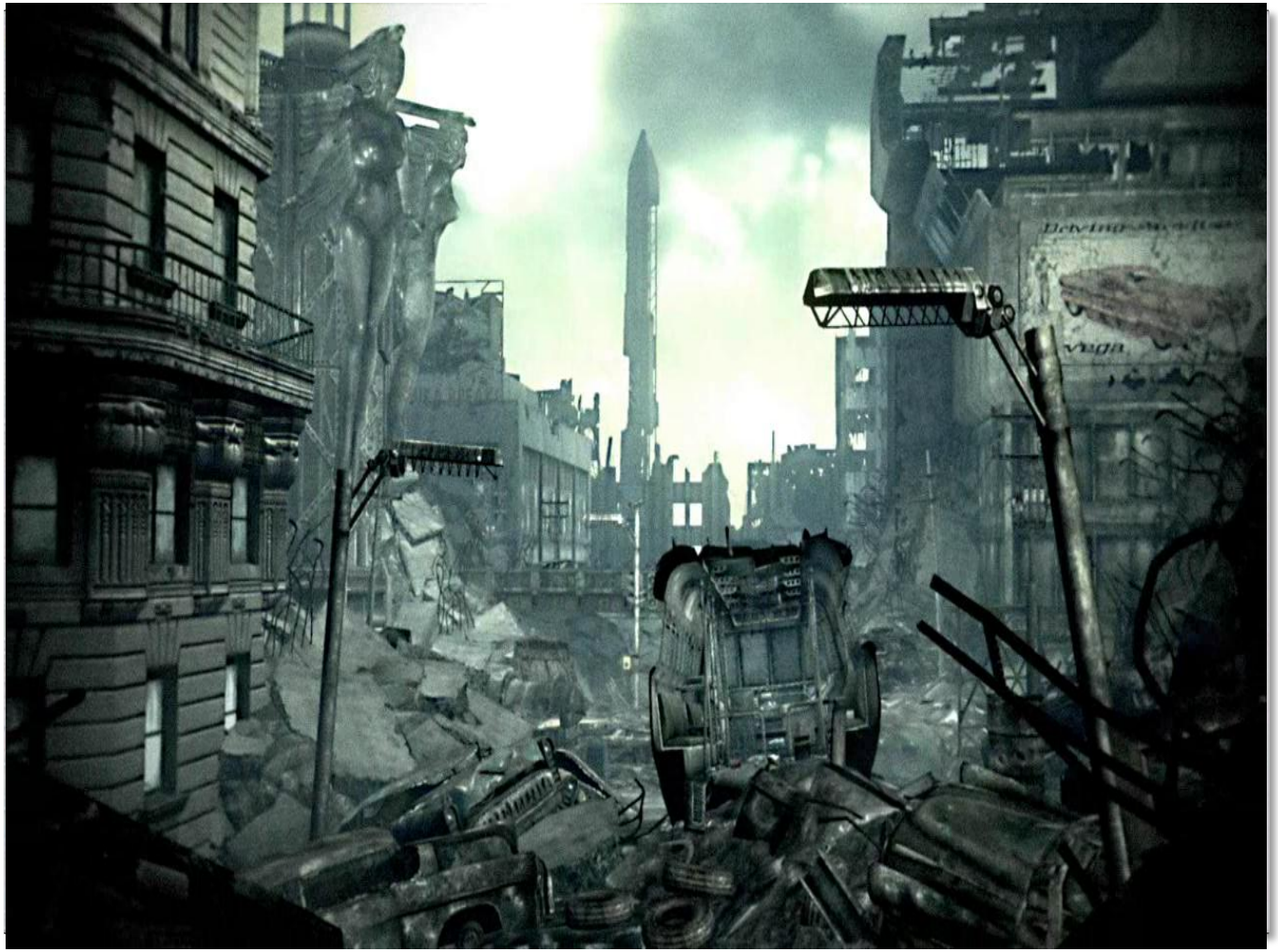
- 2008~ Productivity Innovation Lab, NHN
- 2004~2008 SW Laboratory, Samsung Electronics

❖ **Areas of expertise**

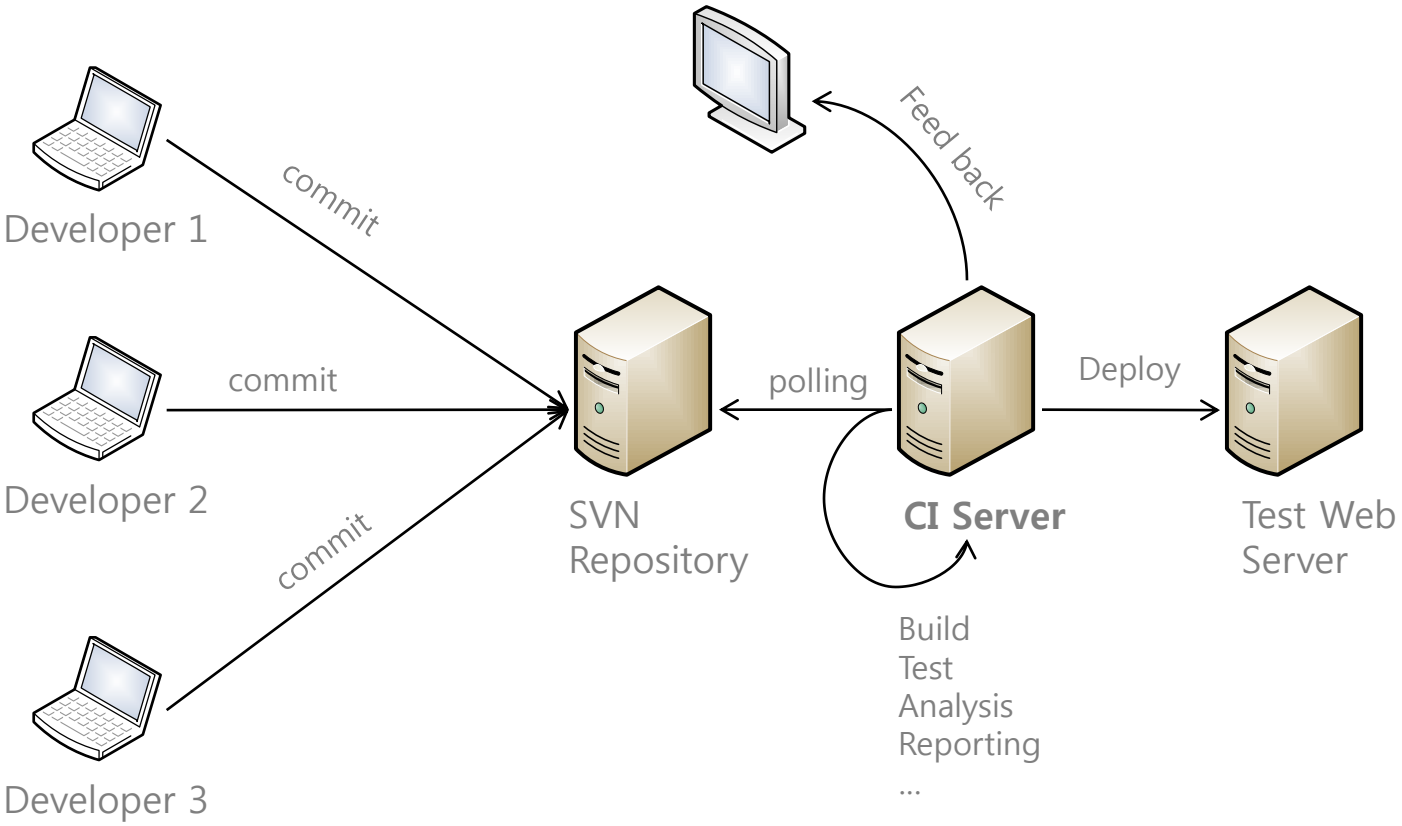
- Software Engineering in SW Development Wide
- Enterprise Web Application Development
- Open Source Project

2. Staged Build

2.1 Broken Window Theory

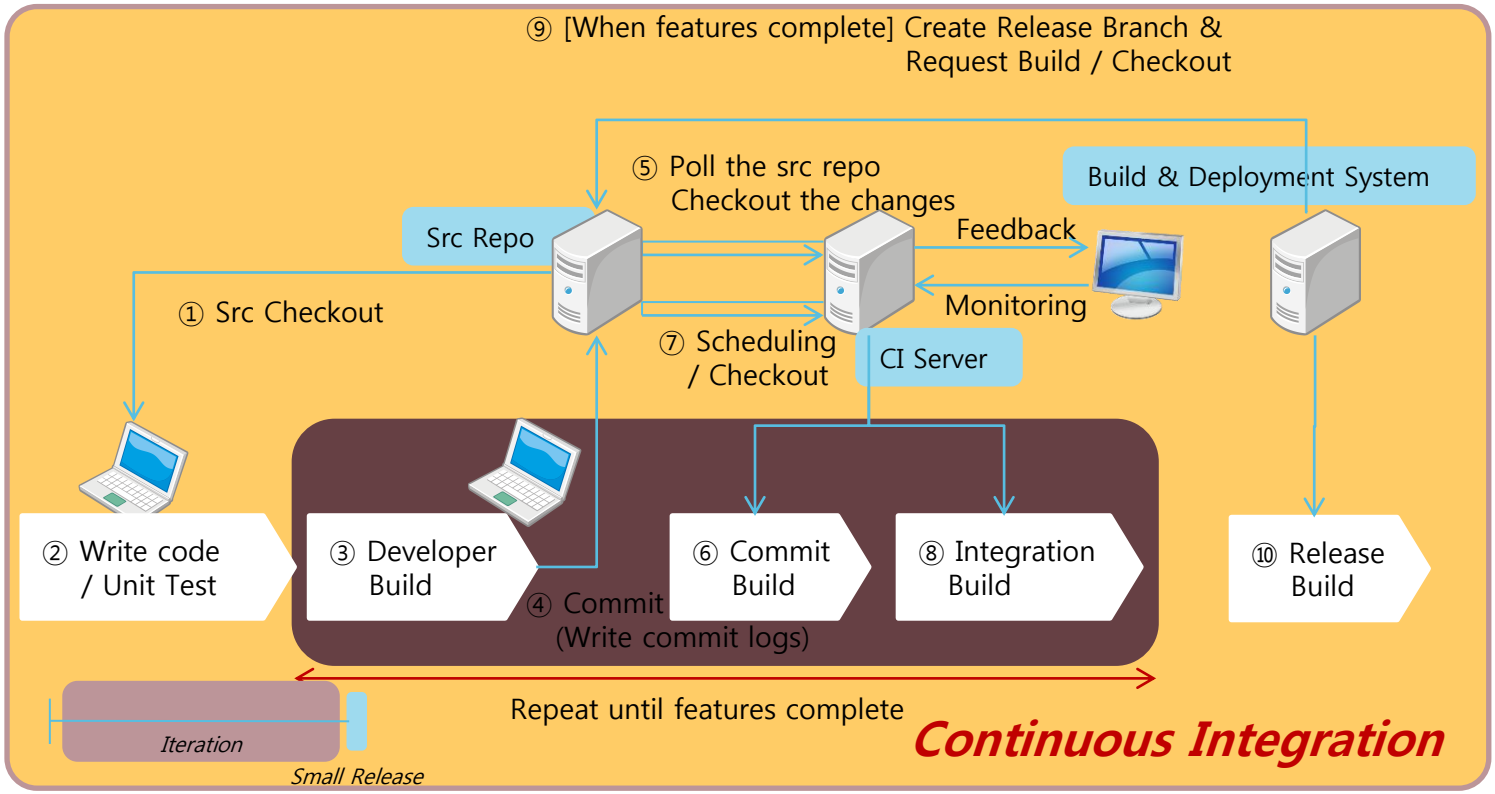


2.2 Continuous Integration(CI)



2.3 Staged Build

Build Integration procedure which is repeatedly performed to release working and tested SW at the end of each iteration until the development finishes
➔ Advanced CI Usage



2.3 Staged Build

Commit Build

Builds which is performed in the separated build server whenever code repository changes occur to detect changes collisions from several developers.

When	• Schedule : Whenever developer commit changes info source repo. • Be able to change pooling timing(10min?) • Duration : within 10 min for build and test.
What	• Detect collision b/w changes from several developers. • Execute all unit tests developed by several developers - Tests which is independent from DB, Platform, Network, - DAO CRUD tests (if necessary) • Perform code reviews on major changes
Post-Condition	• Build success • All passed unit test

2.3 Staged Build

Integration Build

Builds which is performed in the separated build server periodically with more advanced integration test and code analysis

When	• Schedule : Daily(Nightly). Change the interval if necessary. • Duration : with 2 hours
What	• Configure build env and compile • Run unit and integration tests with real server like test environments. - Tests dependent on middleware. - DAO CRUD tests / Automated UI Test - long time taking regression test • Run code analysis - Coding Convention, Code Coverage, Static Analysis, Duplicate Analysis, Cyclomatic Complexity • (When finishing dev scope) Perform dev team own sanity test
Post-Condition	• Build success • Satisfy Quality Practice Criteria - Coding standard conformance rate, code coverage, static analysis defect rate • (When finishing dev scope) More than 90% pass rate of sanity test

2.3 Staged Build

Why separate commit build and integration build

Maximize CI Server operation efficiency

- ✓ Commit build is performed to detect collisions of committed code from several developers. Therefore fast feedback is necessary.
  - Minimum build activity
  - Needs to be real-time.

- ✓ Integration build is performed with full analysis and test to detect more defect behind. It takes time.
  - Maximize build activity
  - Not needs to be real-time.

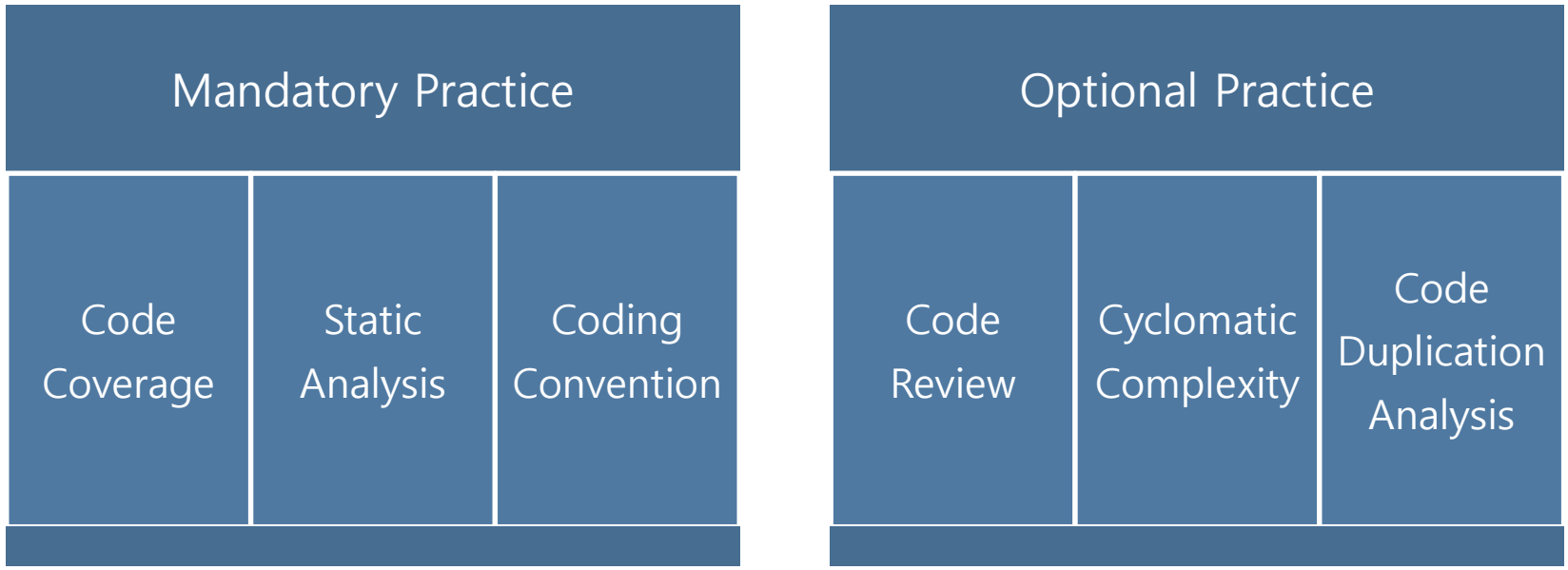
- By separating them,
  Minimize CI server load and Give fast feedback to developers.

2.4 Quality Practices

Mandatory Practices and Optional Practices

❖ Quality Practices

- The minimum quality activities performed by NHN developers to guarantee the defect less SW
- Enforced from 2009



2.4 Quality Practices

Mandatory Practice - Code Coverage

❖ **Should perform developer's automated tests and check the code coverage to improve code quality**

- Test should be measurable and repeatable.
- Test coverages from not only Unit Test but also Integration test can be accumulated.

Metrics	Measures	Red	Yellow	Green	Gold
Statement Code Coverage(%) A/B*100	A. Tested Statements B. Statements	COV<30%	30%≤COV<50%	50%≤COV<70%	70%≤COV
Branch Code Coverage(%) A/B*100	A. Tested Branches B. Branches	COV<20%	20%≤COV<40%	40%≤COV<60%	60%≤COV

2.4 Quality Practices

Mandatory Practice - Static Analysis

❖ **Should remove Static analysis defects found by Static**

- Run static analysis tools and review the defect found.
- Run static analysis tools again when finishing development and Find out the status of defects remained.

Metrics1	Measures	Red	Yellow	Green	Gold
Static Analysis Defect Density (Count/KLOC) A/B*100	A. Weighted count of remained static analysis defect B. Total LOC	6≤Density	4≤Density<6	2≤Density<4	Density<2

2.4 Quality Practices

Mandatory Practice - Coding Convention

❖ **Should confirm NHN Coding Standard to keep the same style in newly created and modified code.**

- Using the Custom NHN checkstyle extensions(Java)
- Using the N'SIQ CppStyle (C/C++)

Metrics	Measures	Red	Yellow	Green	Gold
Coding Standard Conformance Rate (CSCR / %) A/B*100	A. Count of files in which no violation found B. Count of total checked files	CSCR<30%	30%≤CSCR<70%	70%≤CSCR<90%	90%≤CSCR

2.4 Quality Practices

Optional Practice - Code Review

❖ **Review the newly created and modified code**

- How : Choose one of offline or online review
- What : Define criteria which code will be reviewed
- Review code as much as possible. However not mandatory.

Metrics	Measures	Red	Yellow	Green	Gold
Code Review Rate (%) A/B*100	A. Total LOC of files reviewed B. Total LOC of modified or created files	CR<30%	30%≤CR<60%	60%≤CR<80%	80%≤CR

2.4 Quality Practices

Optional Practice - Cyclomatic Complexity

❖ **Identify the complexity code and Reduce complexity**

- Find out complex methods which need to be refactored using tools
- Check the test coverage of complex methods
- Reduce complex methods. However not mandatory.

Metrics	Measures	Red	Yellow	Green	Gold
CC≥30 rate (%) = A/B	A. The count of CC≥30 method B. The count of total methods	Not measured or Not meet org goal	Meet org goal	< 0.5%	= 0%

- Cyclomatic Complexity : Simply the count of if / while / for statements per method

2.4 Quality Practices

Optional Practice - Code Duplication Reduction

❖ **Reduce and Refactor duplicated code.**

- Identify and prioritize highly duplicated code and refactor them with generalization/reuse/abstractions.
- Each project can perform Code Duplication Reduction based on their own decision,
  Code Duplication Reduction is not the mandatory subject to be collected.

❖ **Recommendation**

- Reduce High Prioritized Duplicated code
- Reduce Normal, Low Prioritized Duplicated code by each own decision

High	Normal	Low
50>Duplicated Line	25<Duplicated Line≤50	Duplicated Line≤25

2.4 Quality Practices

Overall Rating Criteria

❖ Code Quality(CQ)?

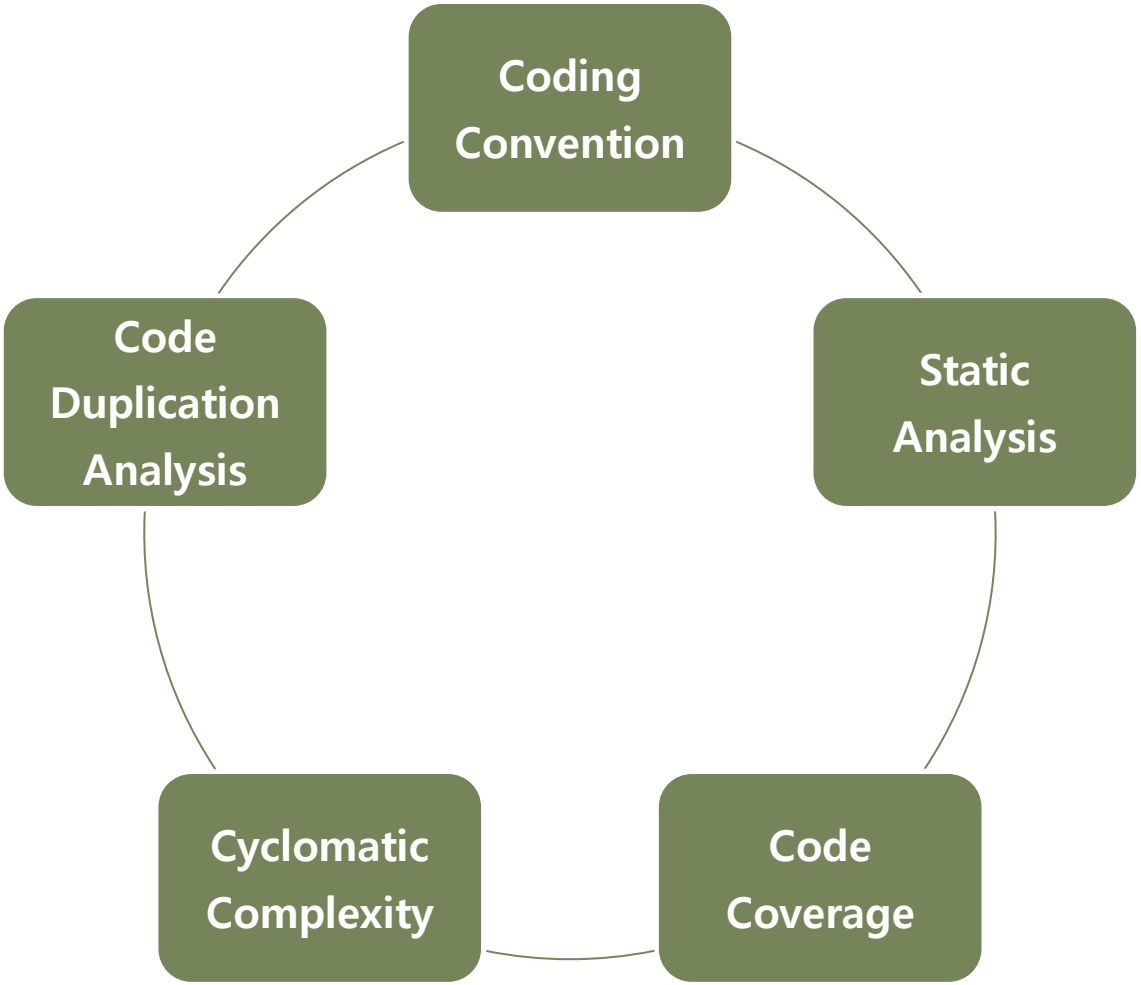
- Indicator to determine the result of Quality Practice performance efficiency
- Varies from development area.

❖ Code Quality Rating Criteria (e.g : Portal service)

Target Quality Metric	Weight	Red	Yellow	Green	Gold
Code Coverage	30	CQ<20	20≤CQ<40	40≤CQ<50	50≤CQ≤70
Coding Standard Conformance Rate	20				
Static Analysis Defect Density	20				

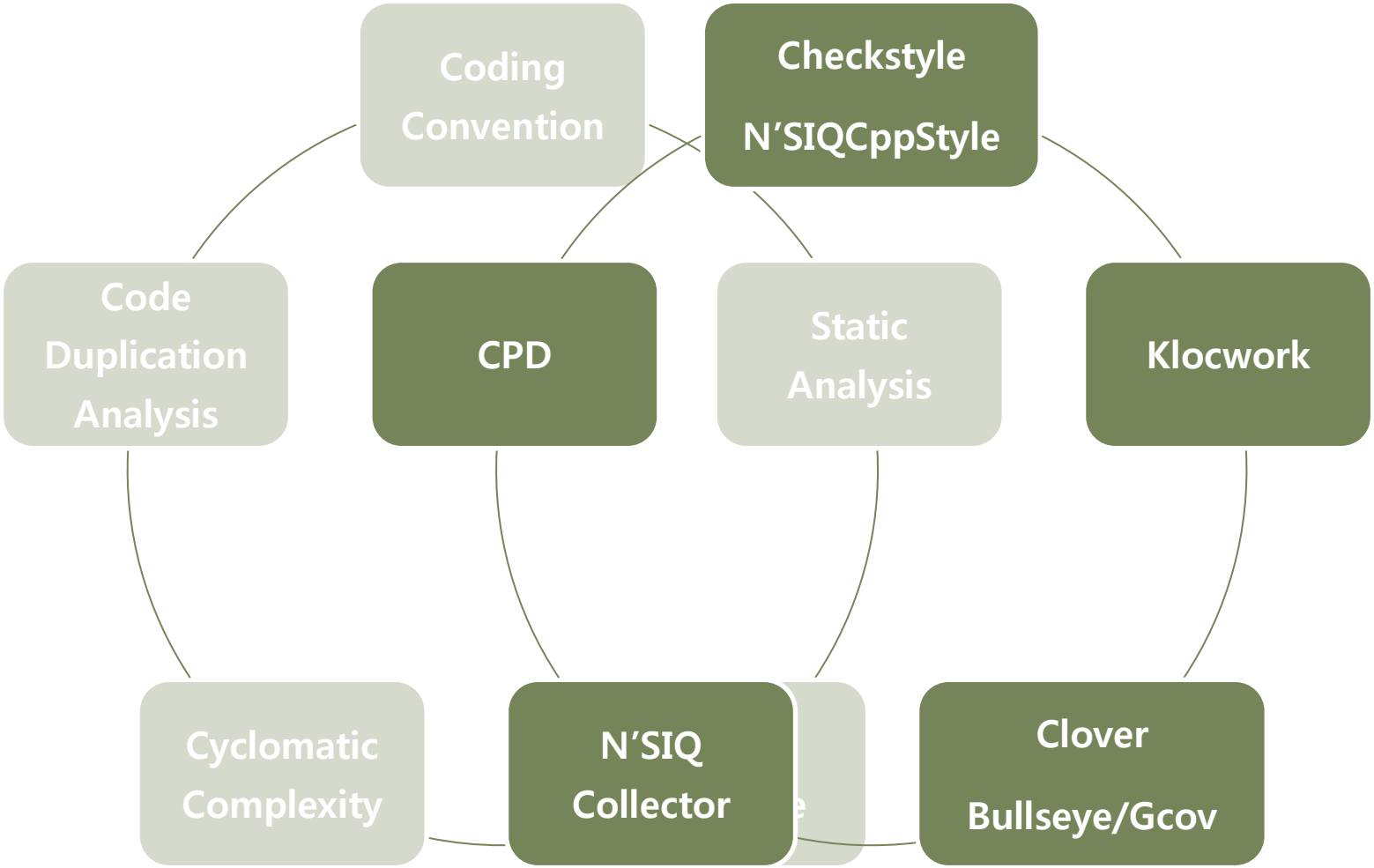
3. Hudson/Jenkins

3.1 Quality Practice on CI



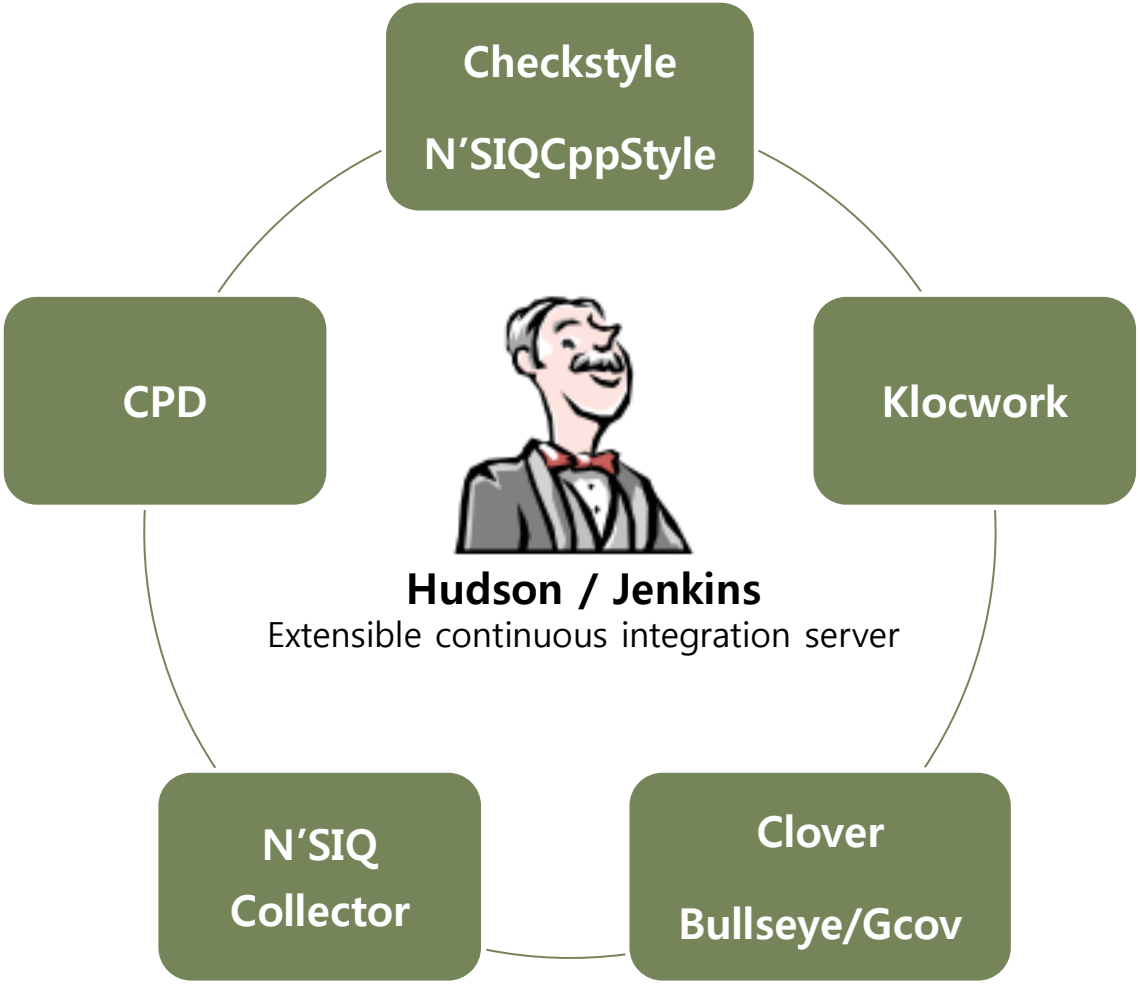
3.2 Quality Practice on CI

Tool support



3.2 Quality Practice on CI
With Hudson

All tools are executed or collected by Jenkins

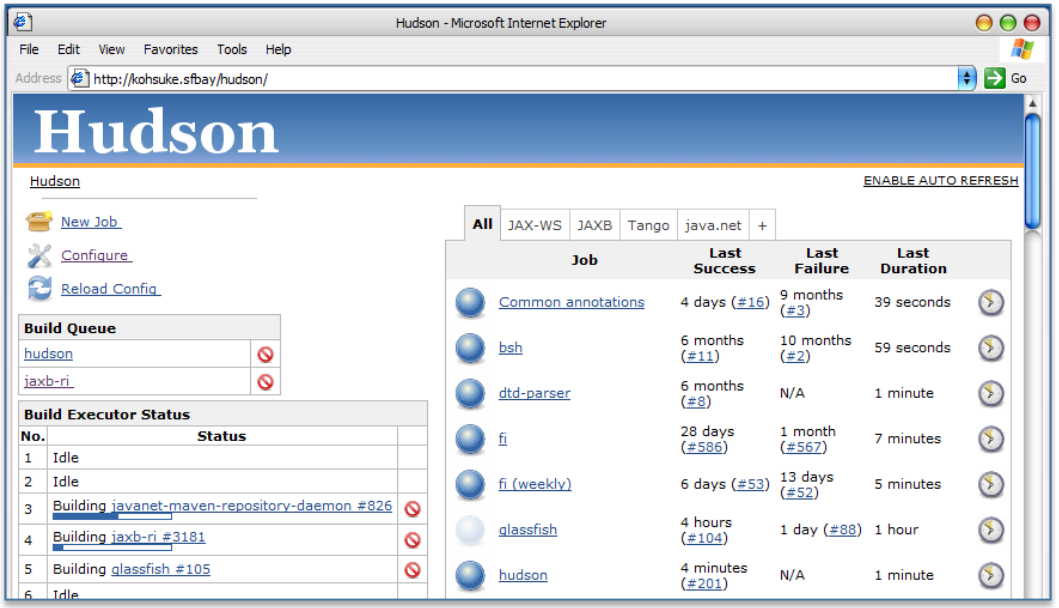


3.3 Hudson Introduction

About

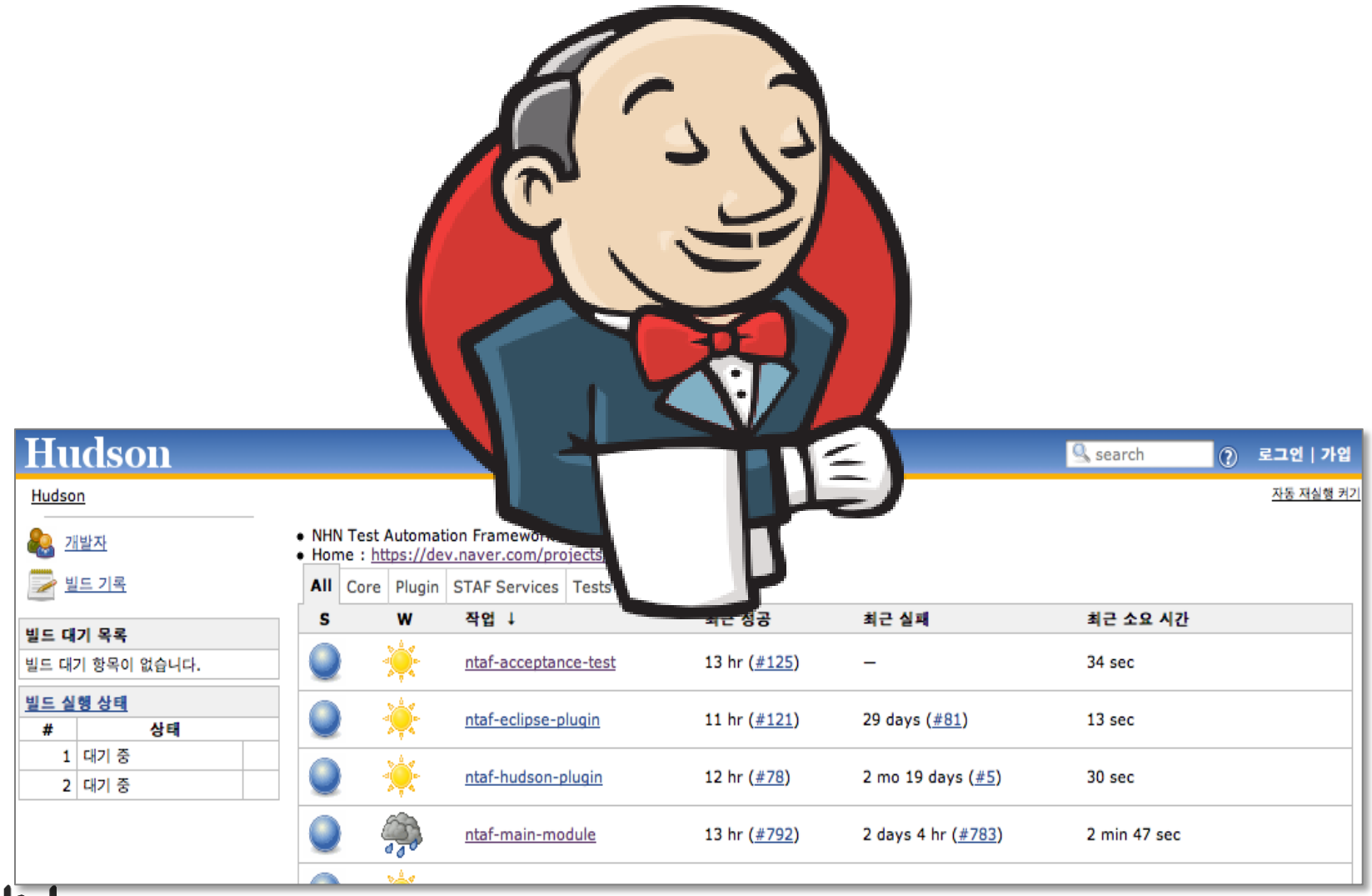
- **Jenkins monitors executions of repeated jobs, such as building a software project or jobs run by cron. Among those things, current Jenkins focuses on the following two jobs:**

- **Current Jenkins focuses on the following two jobs:**
  - Building/testing software projects continuously
  - Monitoring executions of externally-run jobs



3.3 Hudson Introduction

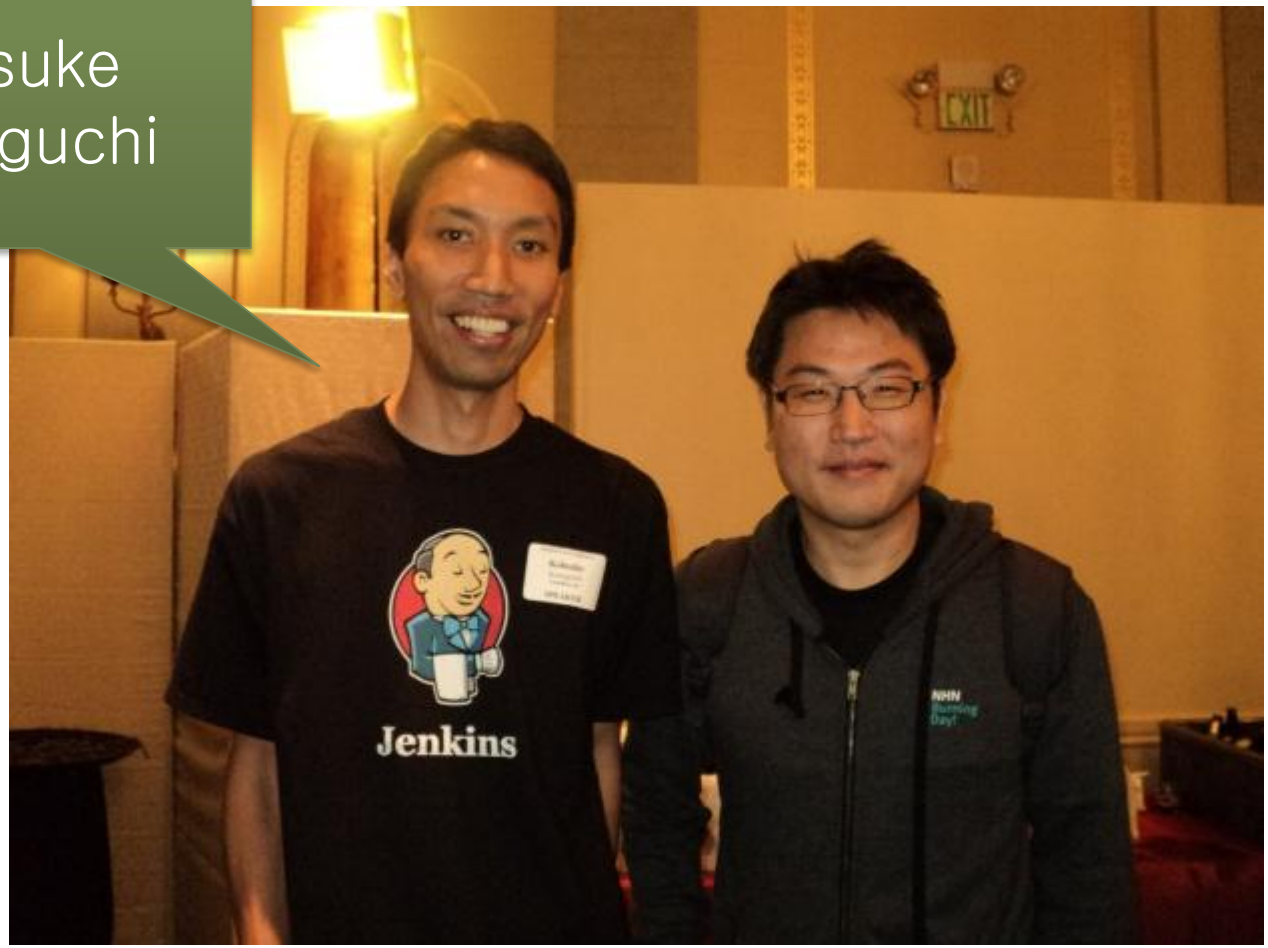
Hudson vs Jenkins



3.3 Hudson Introduction

Jenkins Creator

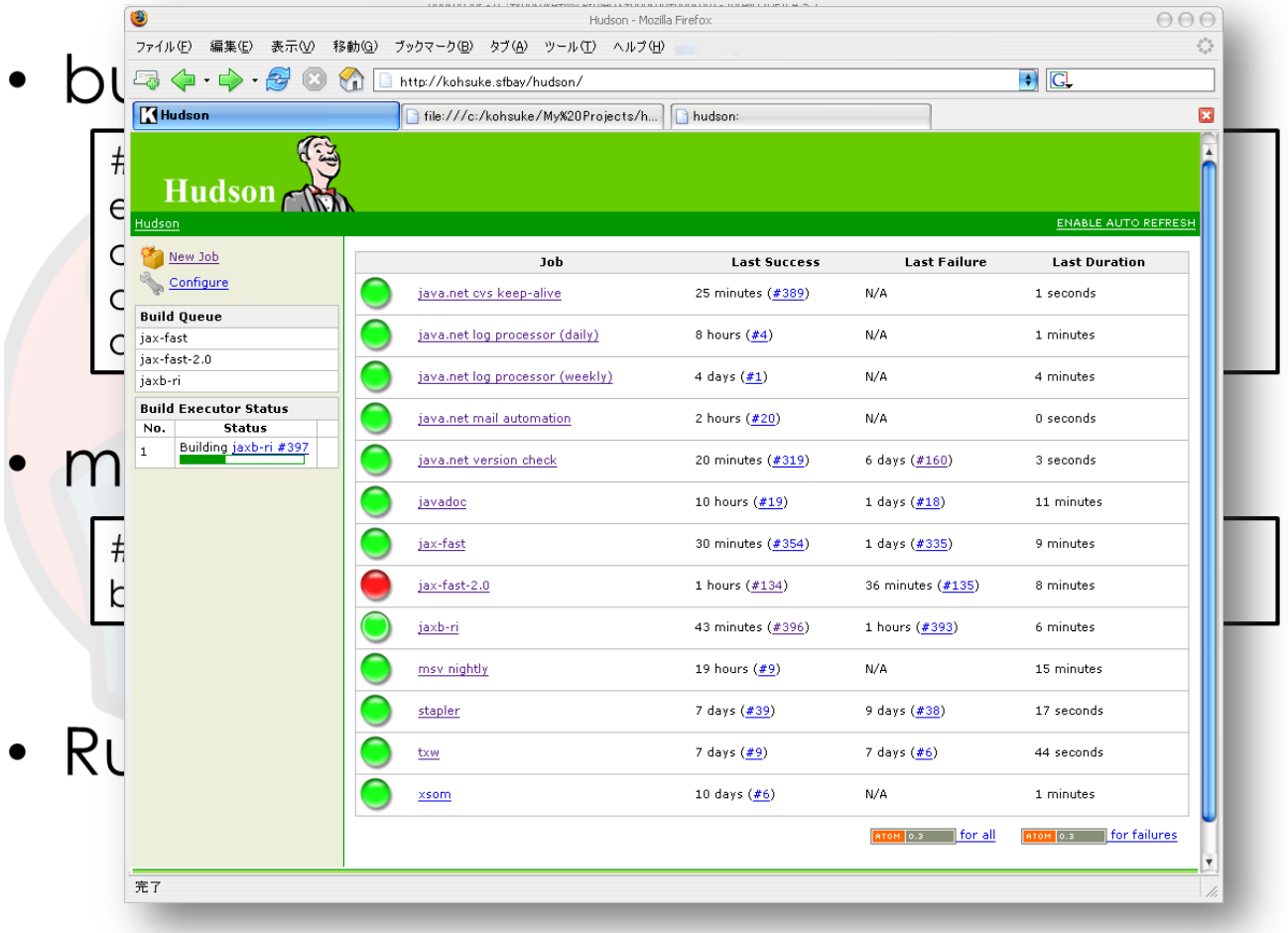
Kyosuke Kawaguchi



3.3 Hudson Introduction

Hudson History

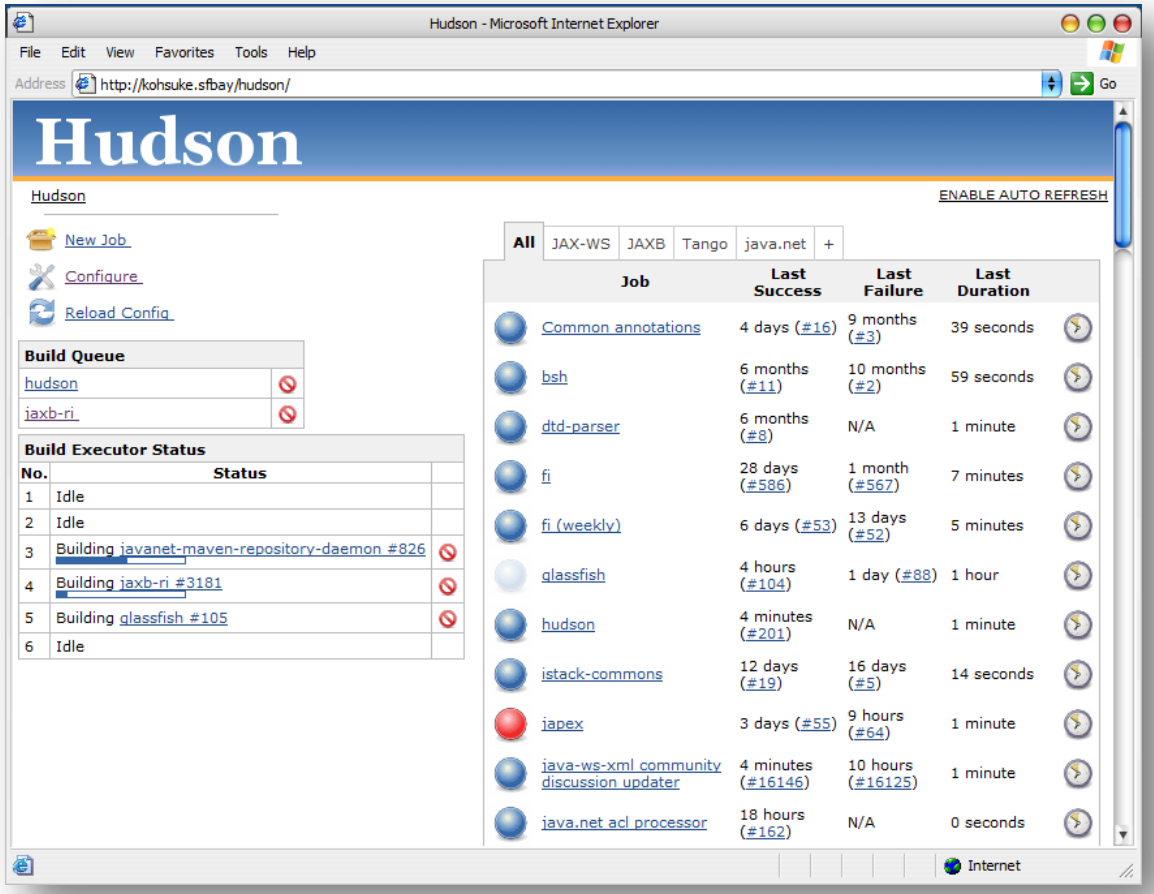
- From Summer, 2004



3.3 Hudson Introduction

Hudson History

- 2006

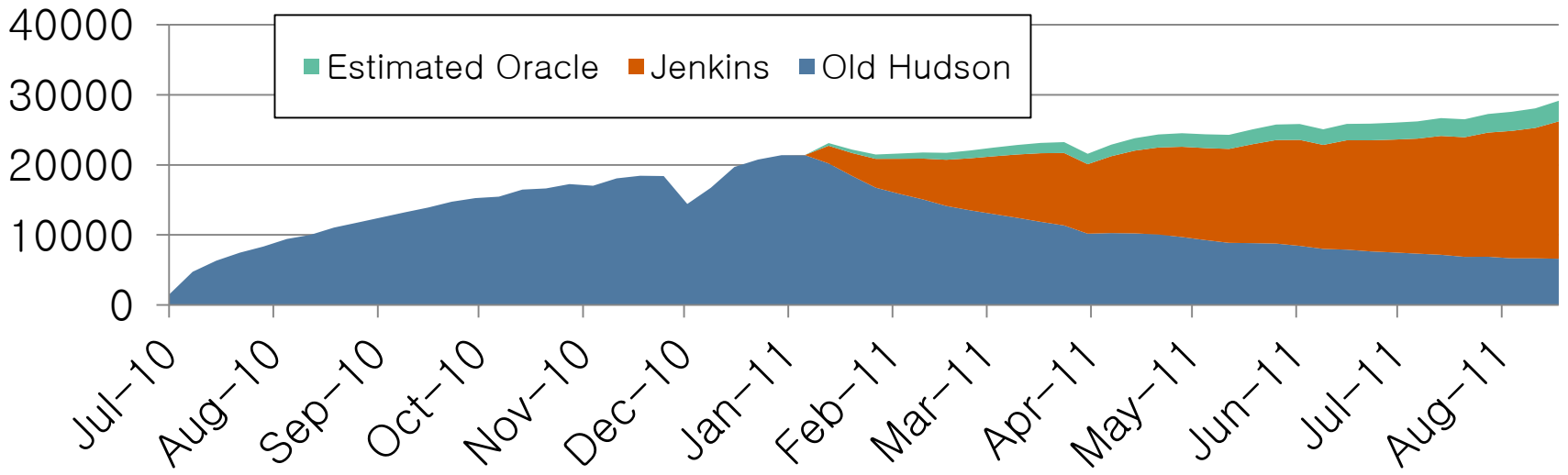


3.3 Hudson Introduction

Jenkins History

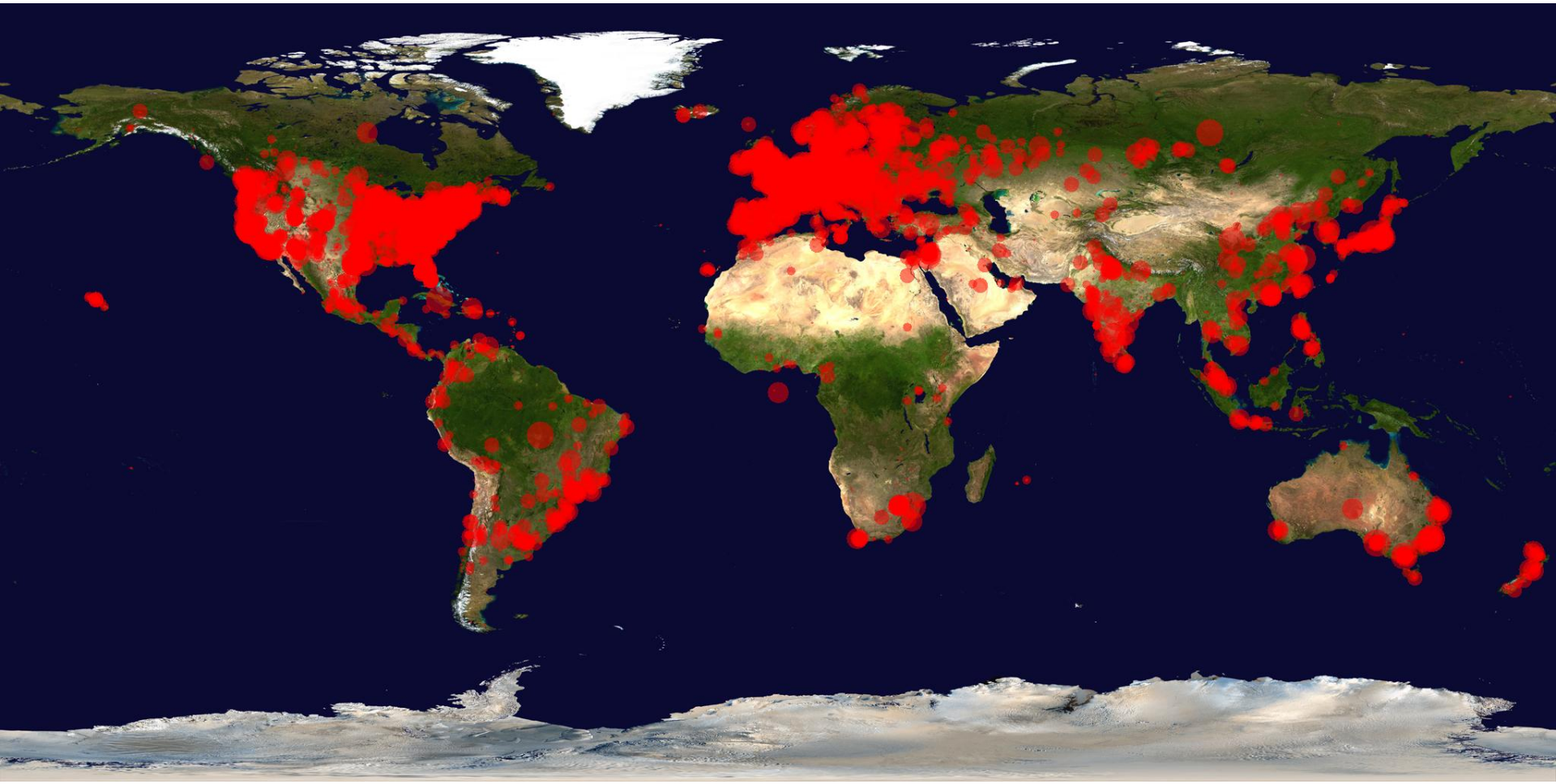
❖ **Jan. 2011. Divorce from Oracle**

- Oracle: "you do it our way or highway"
- Community chose highway: 214 to 14
  - ✓ That's when we became Jenkins



3.3 Hudson Introduction

Usage All Around the World

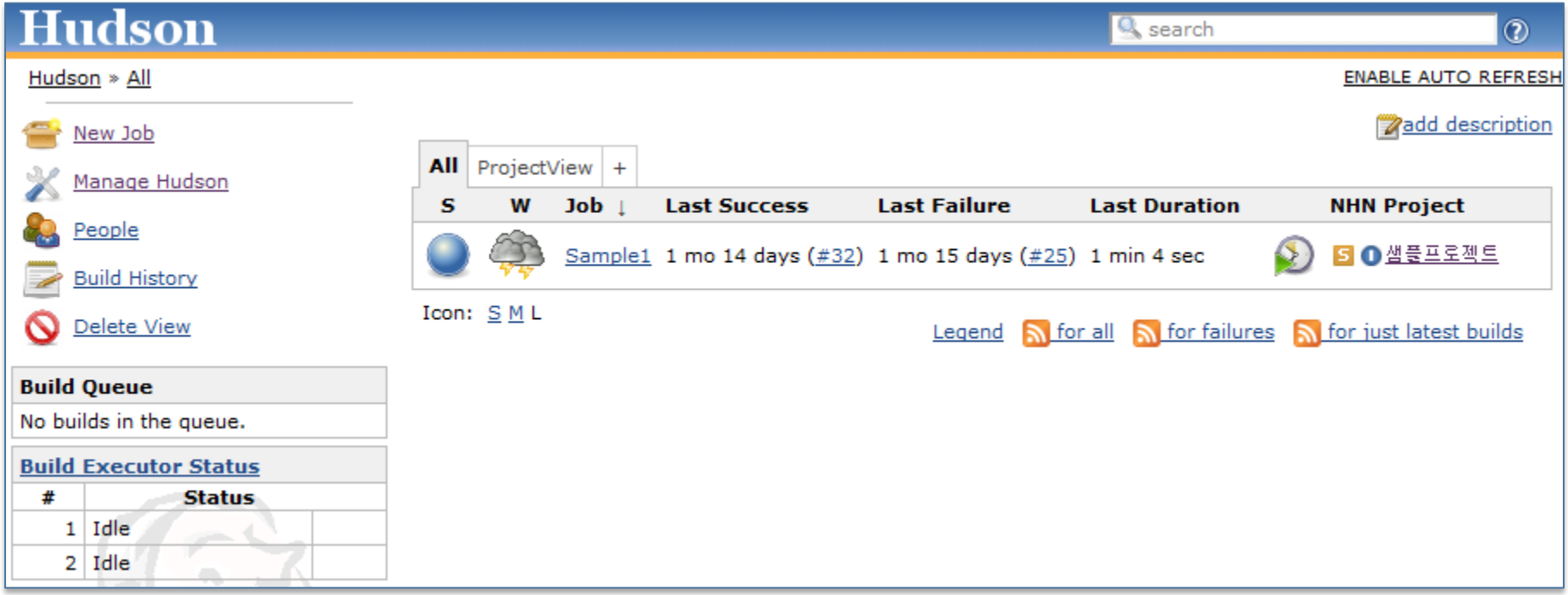


3.4 Hudson Features

- Easy installation
- Easy configuration
- Change set support
- Permanent links
- RSS/E-mail/IM Integration
- After-the-fact tagging
- JUnit/TestNG test reporting
- **Distributed builds**
- File fingerprinting
- **Plugin Support**
- **Easy plugin development environment support**

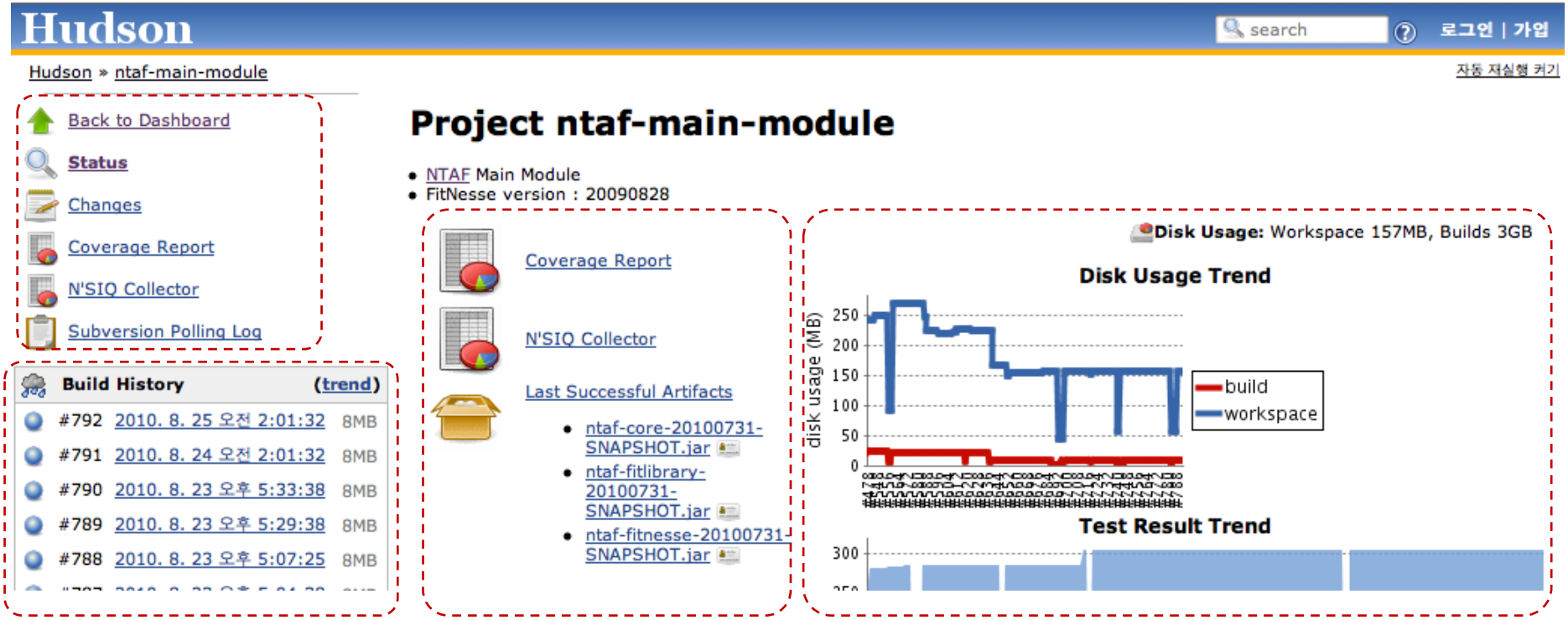
3.4 Hudson Features

Layout - Overall



3.4 Hudson Features

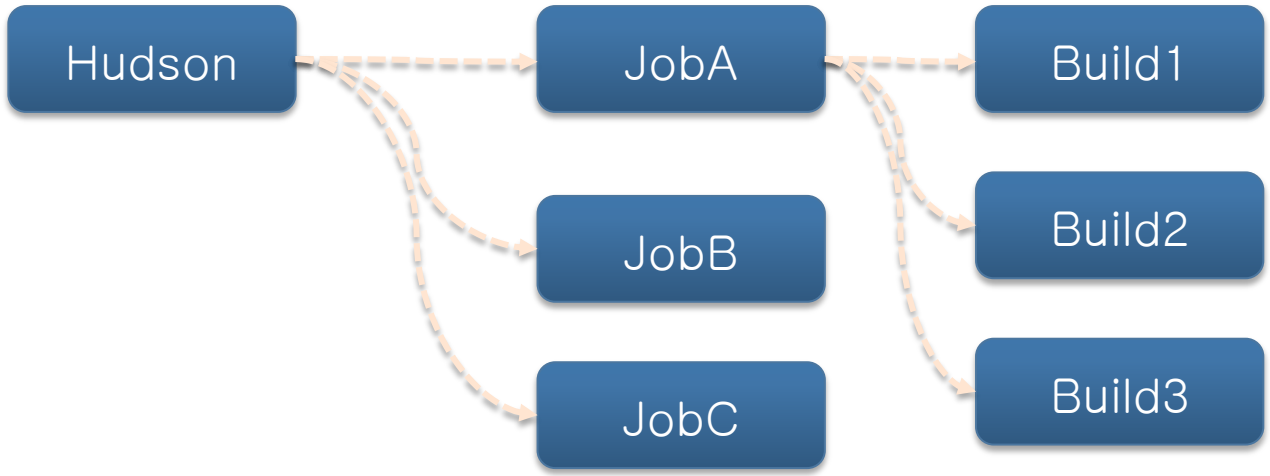
Layout – Project



3.4 Hudson Features

Structural Architecture

- **Conceptual == Physical layer**

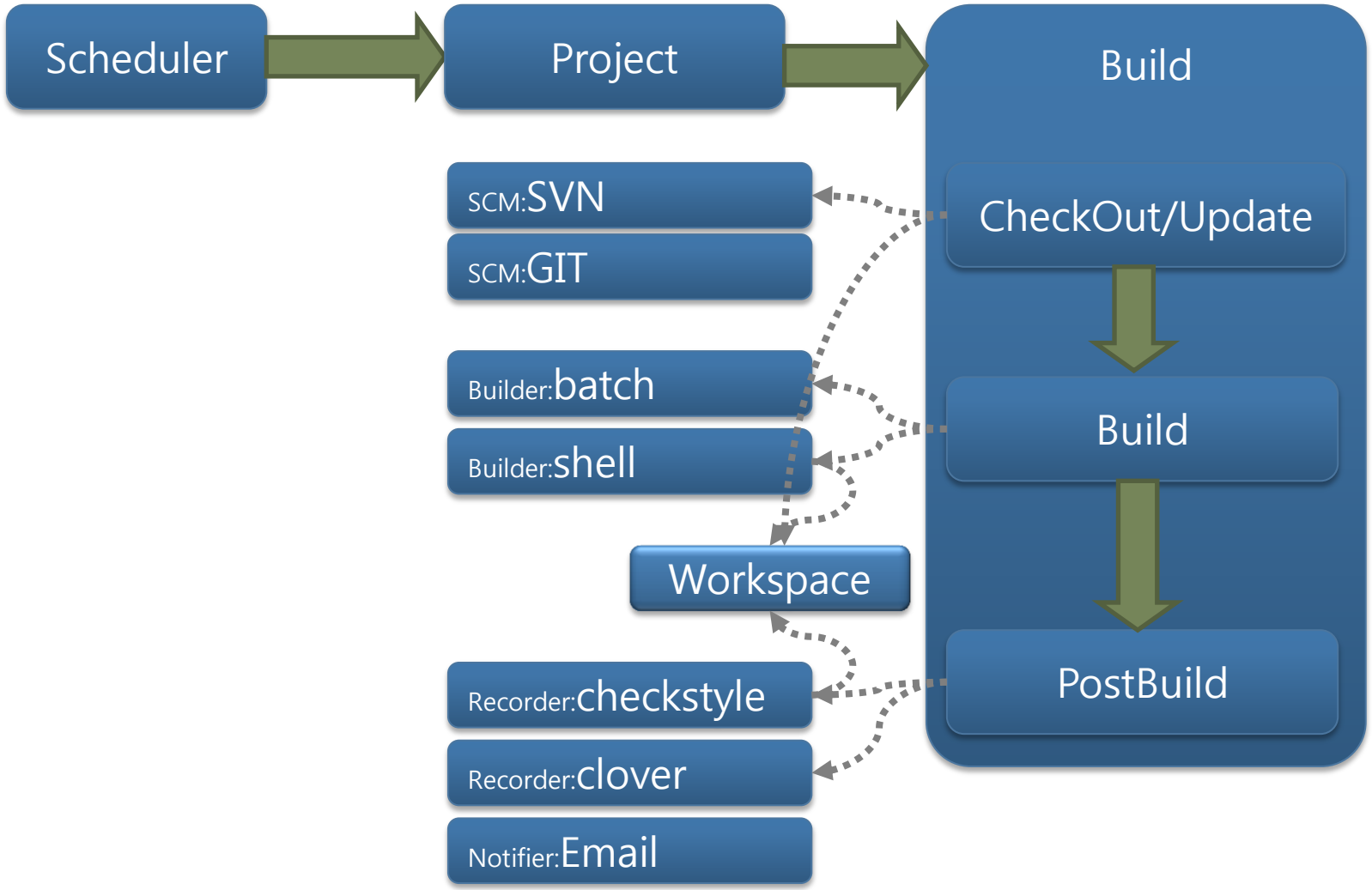


- **Each concept is mapped to Object.**
- **Each object keeps its persistency with XML**

```
Hudson hudson = Hudson.getInstance();
((AbstractProject)hudson.getJob("jobA")).getBuildByNumber(3);
```

3.4 Hudson Features

Runtime Architecture



3.5 Hudson plugins

- Source code management
  - ClearCase Plugin
  - File System SCM Plugin
  - Mercurial Plugin
  - Perforce Plugin
  - Harvest Plugin
  - Team Foundation Server Plugin
  - Template Project Plugin
  - Accurev Plugin
  - CVS Plugin
  - …
- Build triggers
- Build tools
- Build wrappers
- Build notifiers
- Slave launchers and controllers
- Build reports
- Artifact uploaders
- Other post-build actions
- External site/tool integrations…

More than 400 plugins

* For more details refer
http://wiki.hudson-ci.org/display/HUDSON/Plugins

3.5 Hudson plugin

Hudson plugin developed By NHN

- Hudson QD Plugin : Sync collected metric with Quality Dashboard
- Hudson N'SIQ Collector Plugin : Show LOC, Complexity collected by N'SIQCollector
- Hudson Klocwork Plugin : Show static analysis result analyzed by Klocwork
- Hudson NHN Auth Plugin : Make hudson authenticated using MyNEXT ID
- Hudson CovComplPlot Plugin : Show Coverage/Complexity relation graph
- Hudson SimpleUpdateSite Plugin : NHN Custom Hudson Plugin updatesite

3.6 Hudson and QualityDashboard

Manager Needs

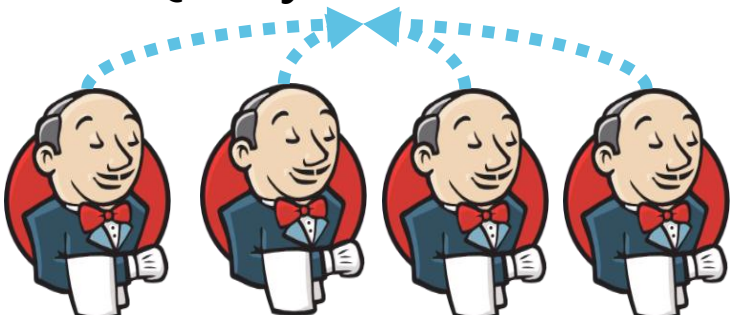
I'm Steve Jobs.
I'd like to see all apple projects status in a big picture.

3.6 Hudson and QualityDashboard

Quality Governance

	45	51	18	10	프로젝
프로젝트명	조회기준일	Code Quality	코딩 표준 준수율 (%)	Code Coverage (Branch) (%)	Code Coverage (Statement) (%)
		?	?	?	?
[동영상서비스] 뮤직	2011.08.30	진행중 0.00			측정안함
통합DB관리시스템	2011.08.22	진행중 52.63	100	측정안함	42.1
Coverage4iBatis	2011.08.22	진행중 55.32	100	85	측정안함
Markupstyle	2011.08.22	진행중 66.32	100	82.1	측정안함
통합DB관리시스템	2011.08.22	52.63	100	측정안함	42.1
[Design Studio] 디자	2011.08.22	진행중 36.41	91.47	0.33	측정안함

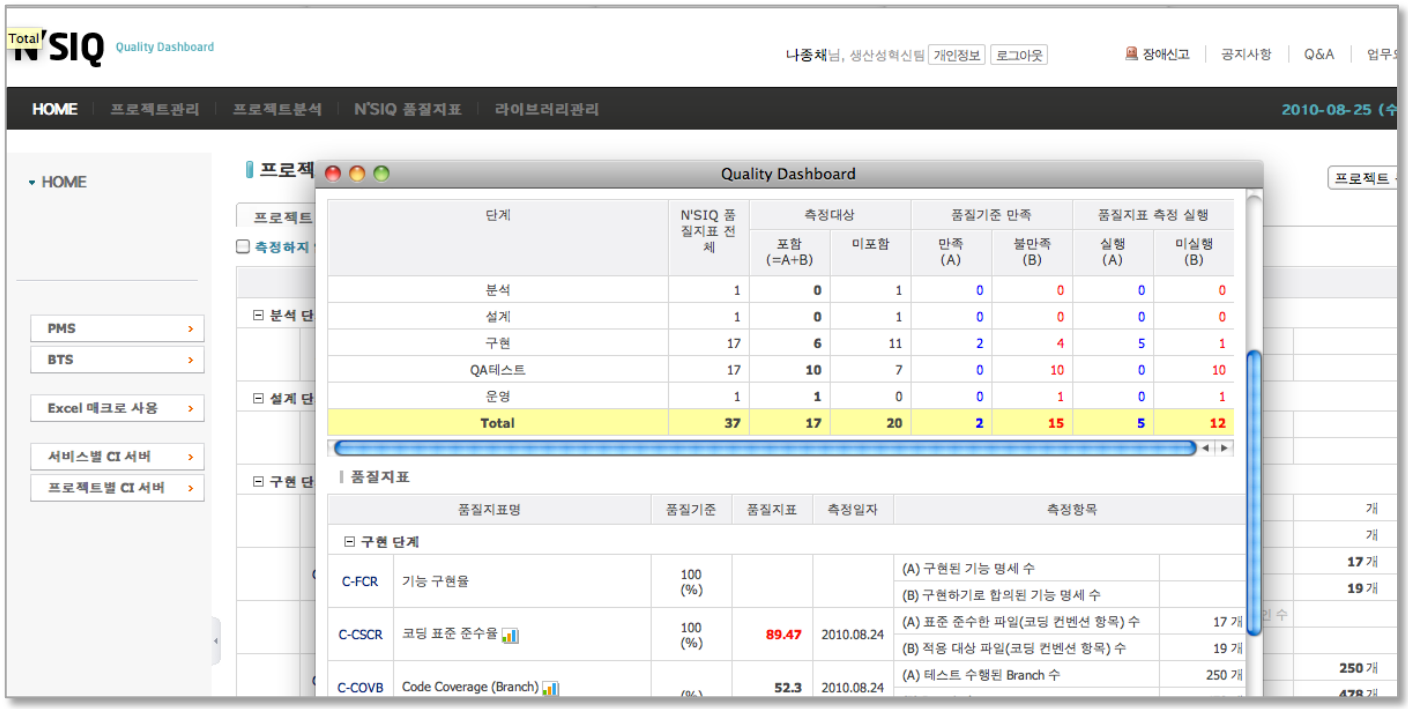
Quality Dashboard



3.6 Hudson and QualityDashboard

Quality Governance

- Qualty Dashboard (http://nsiq.nhncorp.com/)
- Each Hudson Project report its metrics to QD by QD plugin



4. CI Lab

4.1 Install Hudson

Download

- **All Instructions are available in**
  - http://dev.naver.com/projects/hudsonedu/wiki/Java실습스크립트

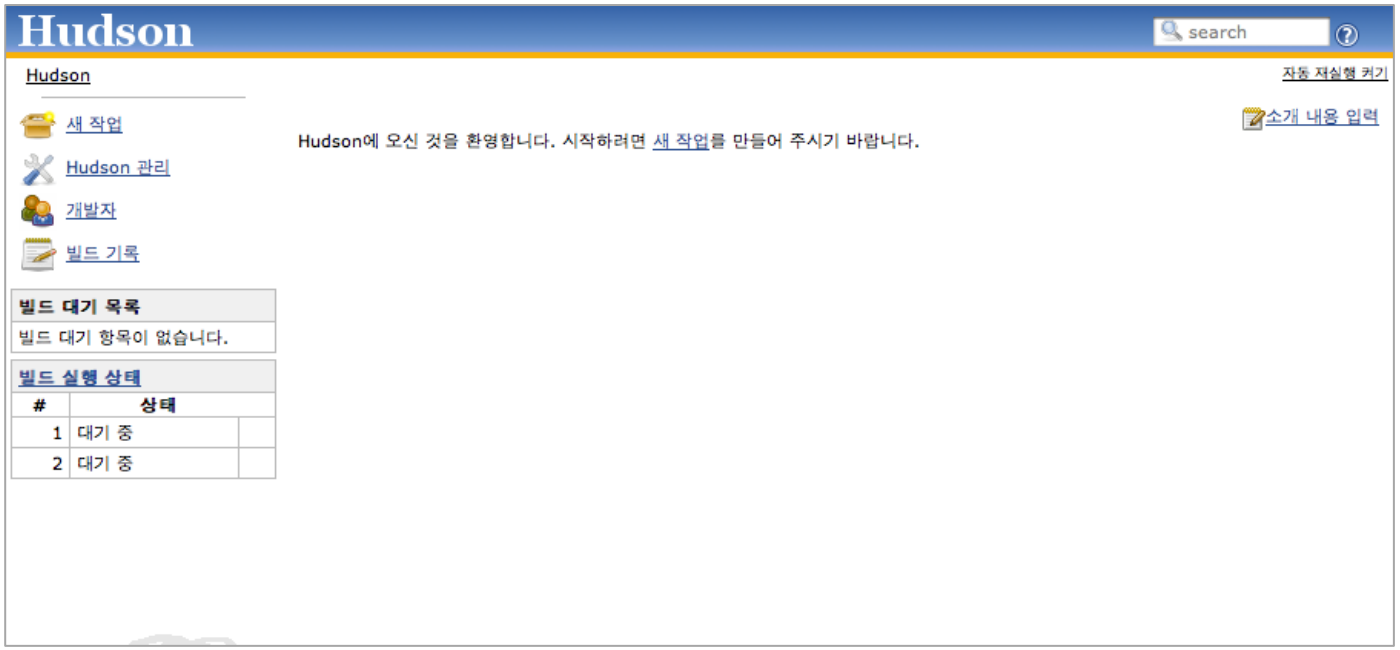
- **Hudson Download**
  - http://hudson-ci.org/downloads/war/
  - Download 1.395.1 version

- **Tomcat Download**
  - http://tomcat.apache.org/ ➔ Tomcat 6.0 ➔ Binary Distributions ➔ Core zip link
  - Download and unzip

4.1 Install Hudson

Startup Hudson

- Put the downloaded hudson.war into {TOMCAT_HOME}/webapps
- Run ${TOMCAT_HOME}/bin/startup.bat(win) or startup.sh (linux)
- Open http://localhost:8080/hudson and see the following page

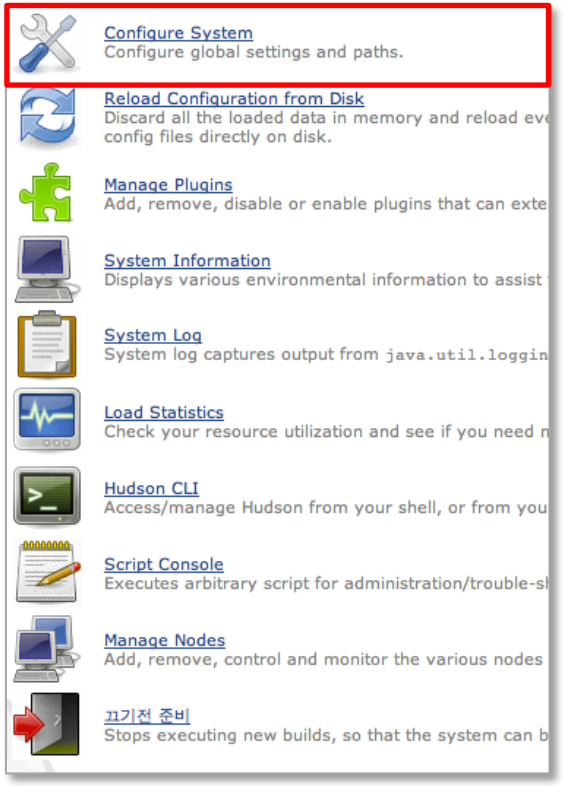


4.1 Install Hudson

Configure System menu

❖ **"Manage Hudson" ➔ "Configure System" : Hudson Global Configuration**

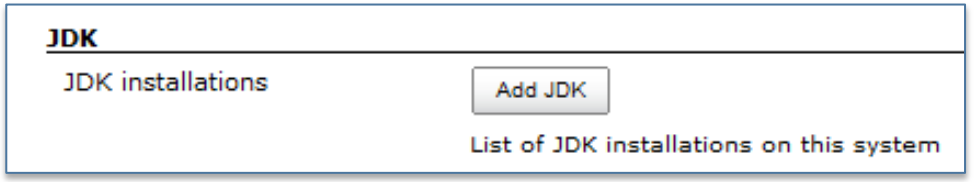
- Setup general Hudson
  env configuration
- Setup Build Tool option
  (Maven, JDK, Ant, Shell) and
  Configure the plugins' global behavior
  - ✓ Automatically installable
    (Ant, Maven, JDK only)



4.1 Install Hudson

Setup JDK

- Click Add JDK



❖ **If JDK is already installed**

1. Uncheck Install automatically
2. Put your own JDK label(e.g : jdk 1.6.0_25) in the "name" field
3. Put JDK path in JAVA_HOME(e.g : C:₩Program Files₩Java₩jdk1.6.0_25)

❖ **If JDK is not installed**

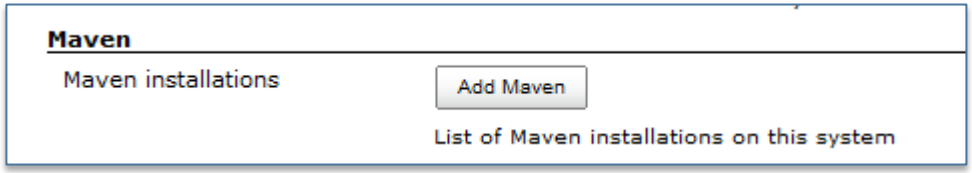
1. Put your own JDK label(e.g : jdk 1.6.0_25) in the "name" field
2. Choose JDK version being installed

❖ **Click "Save" button in the bottom**

4.1 Install Hudson

Setup Maven

- Click Add Maven



❖ **If Maven is already installed**

1. Uncheck Install automatically
2. Put your own Maven label (e.g. Maven 2.2.1) in the “name” field
3. Put Maven path in MAVEN_HOME(e.g. C:₩dev₩apache-maven-2.2.1)

❖ **If Maven is not installed**

1. Put your own Maven label(e.g. Maven 2.2.1) in the “name” field
2. Put Maven path in MAVEN_HOME(e.g. C:₩dev₩apache-maven-2.2.1)

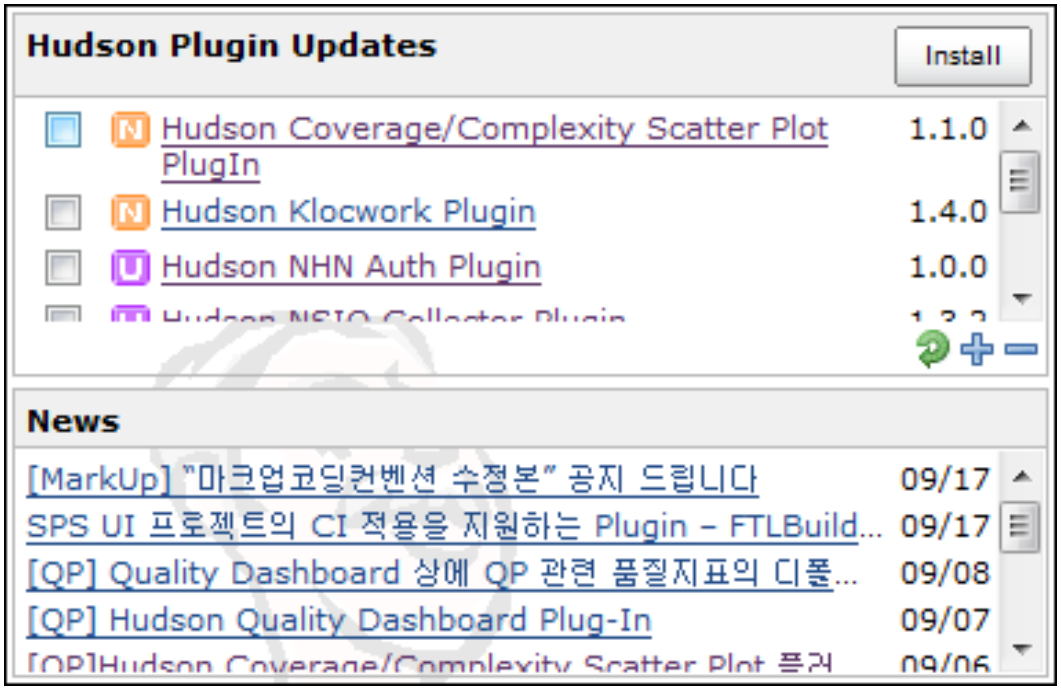
❖ **Click “Save” button in the bottom**

❖ **You can setup Ant same way as Maven**

4.1 Install Hudson

SimpleUpdateSite Plugin

- Enable each Hudson to connect Custom Plugin UpdateSite

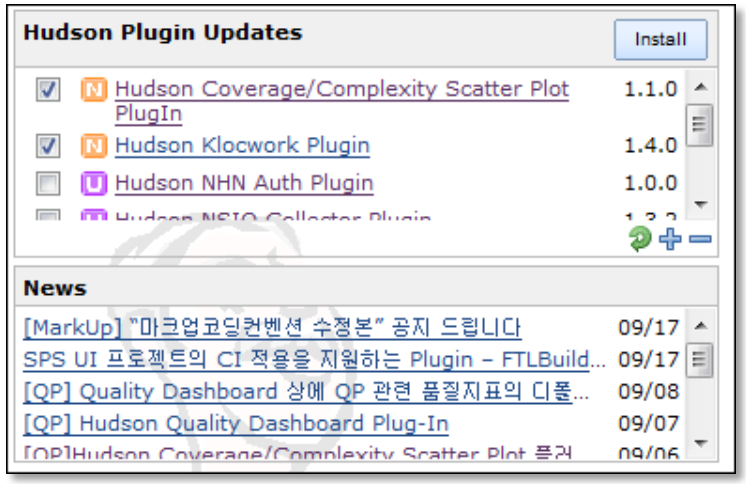


- If you're using Jenkins, you can download it from Jenking update site.

4.1 Install Hudson

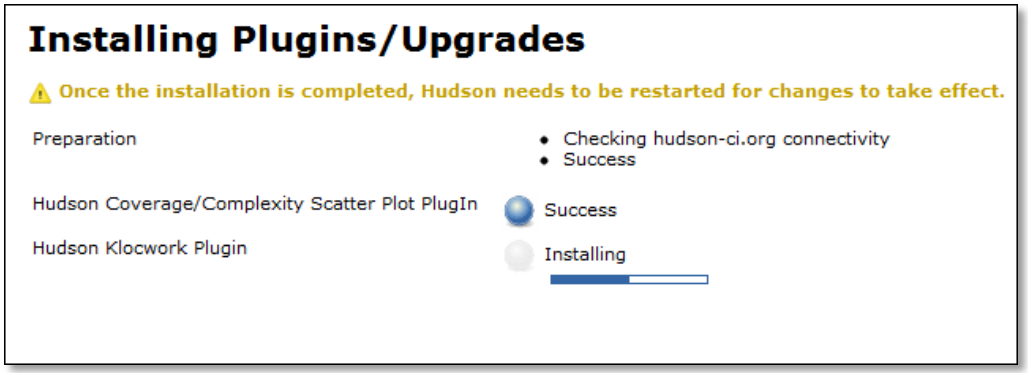
SimpleUpdateSite Plugin

- If you have Admin permission, show following.



1. Select plugins to be installed
   - ✓ N : New plugin
   - ✓ U : Updatable plugin
   - ✓ I : Already installed plugin

2. Click Install button



4.2 Lab Introduction

Step

- Create Hudson Project
- Setup Sample Project Checkout
- Setup Unit Test
- Setup QP Tools

4.2 Lab Introduction

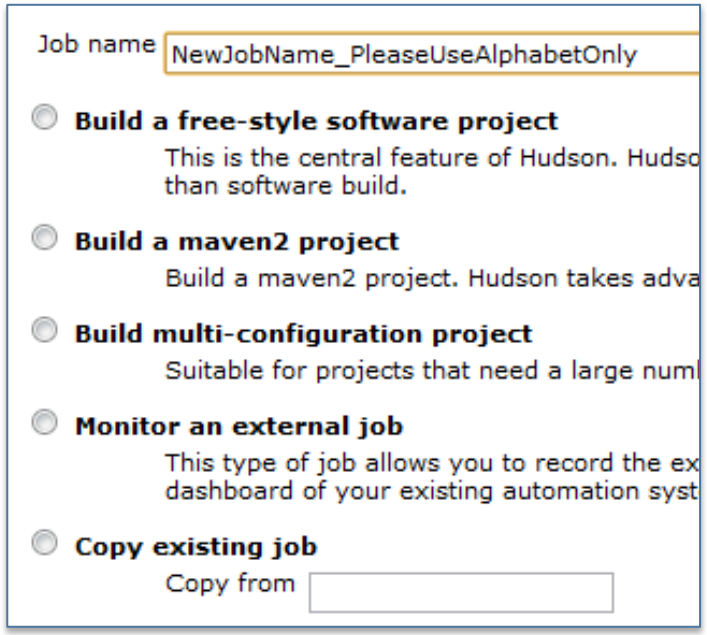
Sample Project

- **Sample project in dev.naver.com**
  - https://dev.naver.com/svn/hudsonedu/trunk

4.3 Configure Hudson Project

Register new Hudson Project (Job)

- **Click "New Job" to register new Hudson Project**



- **Job Name**
  - Should be [a-zA-Z][a-zA-Z0-9_]*
  - Why??
    - ✓ It will be used as the folder name in which code are checked out and built.
- **Select "Build a free-style software project"**

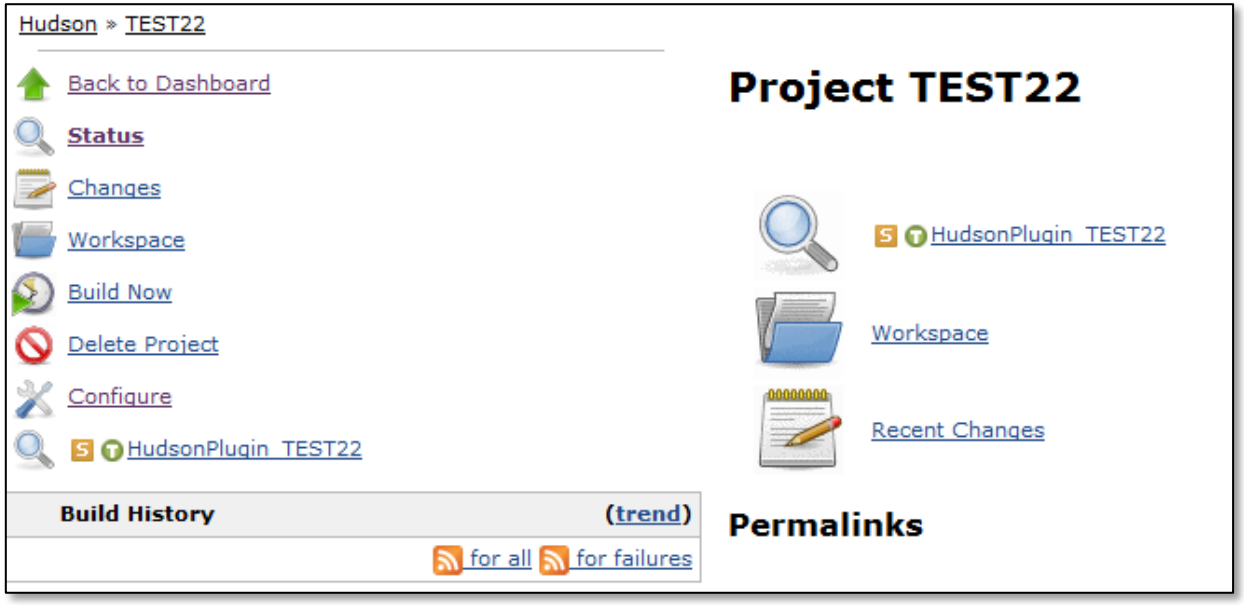
Create Hudson Project with "edu_XX"

4.3 Configure Hudson Project

NHNProject Plugin

❖ **NHNProject Plugin :**

- Provide and display properties which represent project characteristics

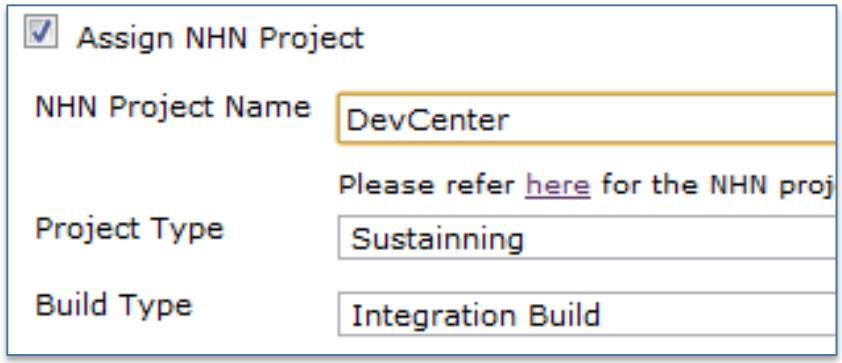


4.3 Configure Hudson Project

NHNProject Plugin

- Installable from SimpleUpdateSite

❖ **How to configure NHNProject plugin**

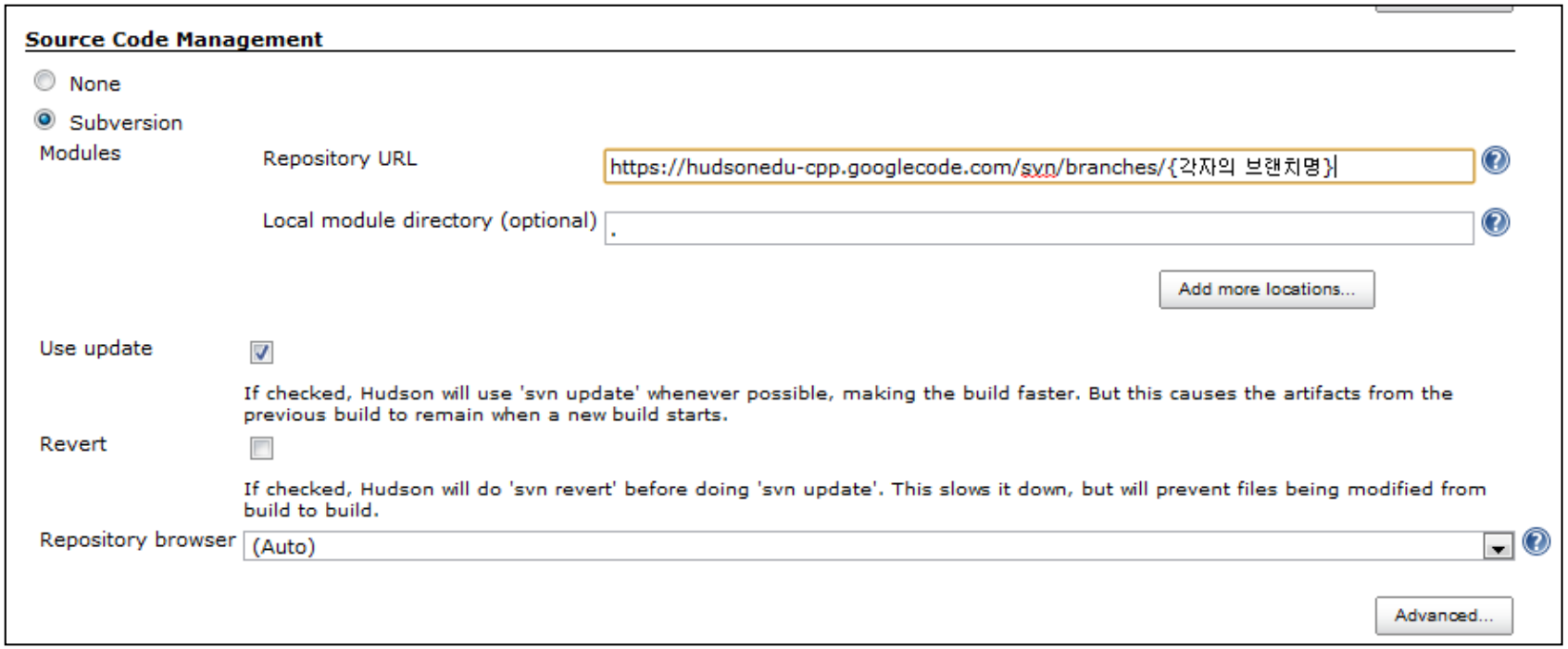


Assign your project name / project type / build type

4.3 Configure Hudson Project

Src Repo

- **Specify source code repository**



Specify your SVN repo

4.3 Configure Hudson Project

Build Trigger

❖ **Build Triggers : Set up the Hudson build start event**

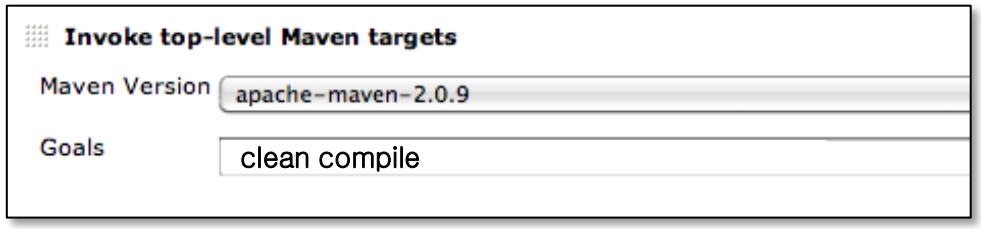
- Build after other projects are build
- Build periodically
- Poll SCM : Execute build when detecting source code changes(commit)
- Cron expression

```
Min Hour Date Month Week
E.g)
    Every minute → * * * * *
    Every 5 min → */5 * * * *
```

4.3 Configure Hudson Project

Add Builders

- Build : Execute build command for various executor
  - ✓ Execute shell(Linux)
  - ✓ Invoke top-level Maven targets
    - Put maven goals necessary to build maven project
    - E.g)



  - ✓ Execute Windows batch command (Windows)
  - ✓ Invoke Ant : Execute ant target on ant build script
- Post-build Actions : Define tasks after build
  - ✓ Mostly import generated doc(e.g: xml) to Hudson for display

Specify maven goals (clean compile)

4.3 Configure Hudson Project

Build Now

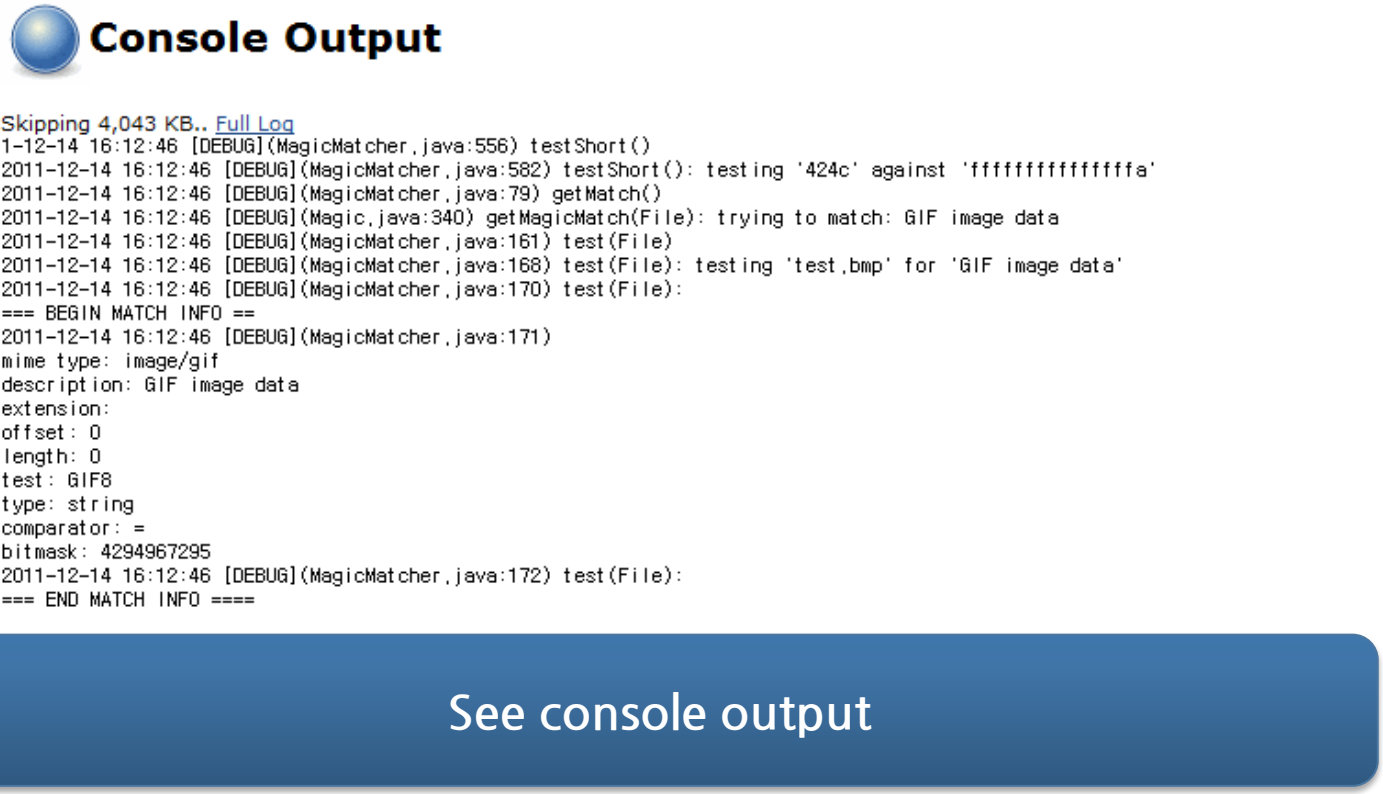
- **Not only scheduled build But also Direct build**
  - Click "Build Now" on left panel
  - Show up new build with sequence in build history
  - Can check build status in Build History
    - ✓ Success Build 
    - ✓ Unstable Build 
    - ✓ Failed Build 
    - ✓ Canceled Build 

Click Build Now

4.3 Configure Hudson Project

Build Log

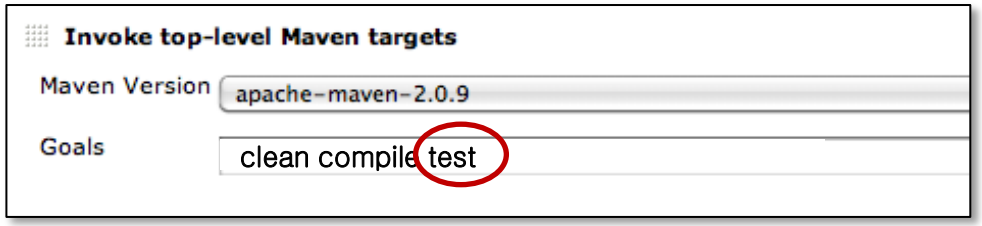
- **Hudson shows build log almost realtime.**
  - Click Build in Build panel and Click Console Output



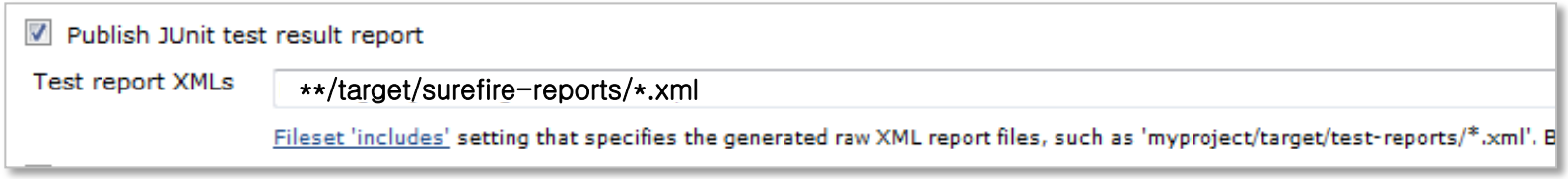
4.4 Enable Unit Test

Setup Unit Test Execution

- **supports JUnit in nature for test result display**
  - Click configure button in left panel
  - Add test goal in "Invoke top-level Maven targets"

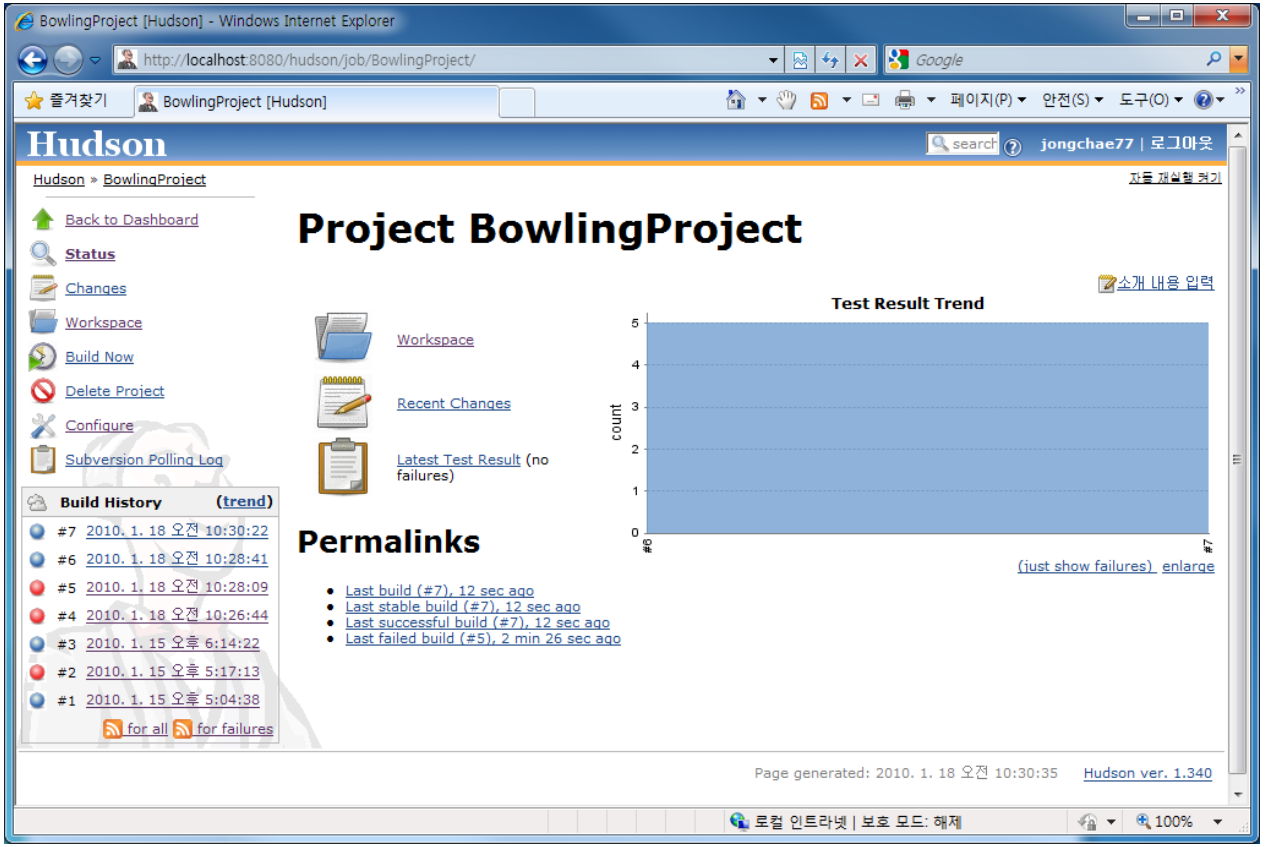


  - Click "Publish JUnit test result report" in Post-build Actions
  - Input result result xml file location)



4.4 Enable Unit Test

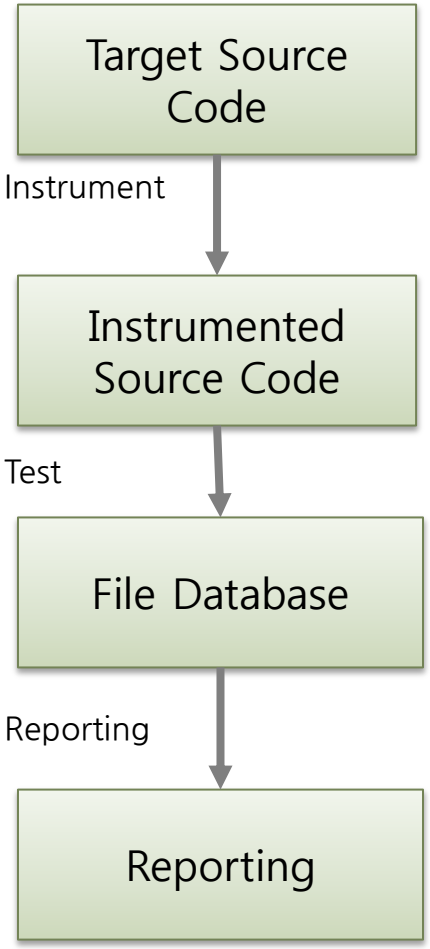
Unit Test Result



Specify maven goals and enable “publish JUnit test result”

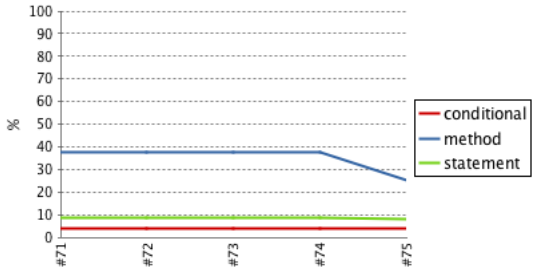
4.5 Code Coverage with Clover

How Code Coverage works




```
public class HelloWorld {
    public static String nullPossibleString = null;

    public static void main(final String[] args) {
        System.out.println(helloWorld());

        if (nullPossibleString.equals(null)) {
            System.out.println("nullPossibleString is null.");
        }
    }

    public static String helloWorld() {
        return hello() + " " + world();
    }
```

```
public class HelloWorld {public static class __CLR3_0_24n4ngbimreni{publi
    public static String nullPossibleString = null;

    public static void main(final String[] args) {try{__CLR3_0_24n4ngbimr
        __CLR3_0_24n4ngbimreni.R.inc(168);System.out.println(helloWorld()

        __CLR3_0_24n4ngbimreni.R.inc(169);if ((((nullPossibleString.equal
            __CLR3_0_24n4ngbimreni.R.inc(172);System.out.println("nullPos
        }
    }}finally{__CLR3_0_24n4ngbimreni.R.flushNeeded();}}

    public static String helloWorld() {try{__CLR3_0_24n4ngbimreni.R.inc(1
        __CLR3_0_24n4ngbimreni.R.inc(174);return hello() + " " + world();
    }finally{__CLR3_0_24n4ngbimreni.R.flushNeeded();}}
```



4.5 Code Coverage with Clover

Various Code Coverage

- Function(method) coverage
- Statement(line) coverage
- Decision(branch) coverage
- Condition coverage – coverage for boolean sub expression
- Condition / decision coverage – Decision + Condition Coverage

❖ **Mandatory to measure Statement or Branch(Decision) coverage in NHN**

❖ **Method > Branch > Statement >= Condition(?) > Condition / decision**

http://en.wikipedia.org/wiki/Code_coverage

4.5 Code Coverage with Clover

Question

```
public int foo(int x, int y) {
     int z = y;

     if ((x > 5) && (y > 0)) {
          z = x;
     }

     return x * z;
}
```

```
assertEquals(49, foo.foo(7, 1));
```

- Branch coverage?
- Conditional coverage?
- Statement coverage?

4.5 Code Coverage with Clover

Modify maven build script

❖ Add clover plugin maven repo in pom.xml

```
<pluginRepository>
        <id>atlassian-m2-repository</id>
        <name>Atlassian Maven 2.x Repository</name>
        <url>http://repository.atlassian.com/maven2</url>
</pluginRepository>
```

❖ Caution!!

- Clover 3.0.2 is not compatible with Maven 3.X
- Clover 3.0.4 has lots of bugs
- Do not install project instrumented by clover into .m2 folder

4.5 Code Coverage with Clover

Modify maven build script

❖ Add Clover build plugin in pom.xml

- Put following lines into pom.xml

```
<build>
  <plugins>
      <plugin>
        <groupId>com.atlassian.maven.plugins</groupId>
        <artifactId>maven-clover2-plugin</artifactId>
        <version>3.0.2</version>
        <configuration>
          <licenseLocation>../../clover.license</licenseLocation>
          <generateHtml>true</generateHtml>
          <generateXml>true</generateXml>
          <generatePdf>false</generatePdf>
        </configuration>
      </plugin>
  </plugins>
</build>
```

4.5 Code Coverage with Clover

Modify maven build script

❖ **Add Clover reporting plugin in pom.xml**

```
<reporting>
    <plugins>
        <plugin>
            <groupId>com.atlassian.maven.plugins</groupId>
            <artifactId>maven-clover2-plugin</artifactId>
            <version>3.0.2</version>
            <configuration>
                <includesTestSourceRoots>true</includesTestSourceRoots>
                <licenseLocation>../../clover.license</licenseLocation>
                <encoding>UTF-8</encoding>
                <jdk>1.5</jdk>
            </configuration>
        </plugin>
    </plugins>
</reporting>
```

Modify your pom.xml

4.5 Code Coverage with Clover

Maven goals

❖ Clover2-Maven-Plugin Goal

Goal	Description
clover2:clean	Initialize Clover Database
clover2:setup	Initialize clover instrumentation feature
test	Run JUnit Test
clover2:clover	Make coverage report under ./target/site/clover folder

4.5 Code Coverage with Clover

Hudson Clover Plugin

❖ Install Hudson Clover Plugin

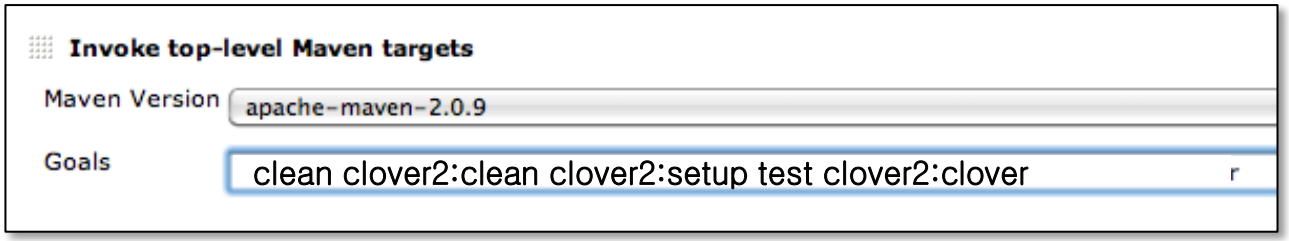
1. Manage Hudson ➔ Manage Plugins ➔ Available Tab
2. Select Clover Plugin and Click install button
3. Restart Tomcat

4.5 Code Coverage with Clover

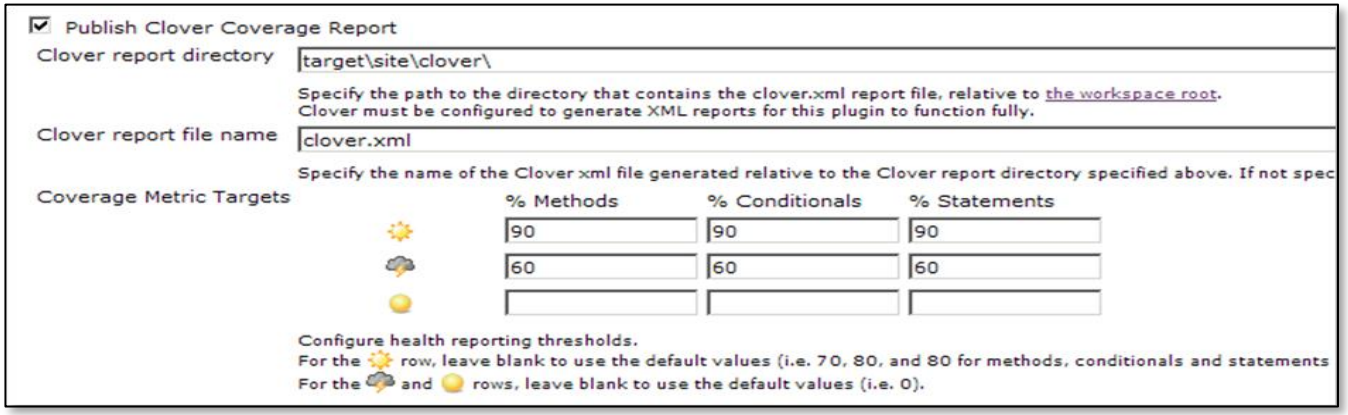
Hudson Clover Plugin

❖ Setup Hudson Clover Plugin per Project

- Add Clover Goal into maven project



- Enable "Publish Clover Coverage Report" and setup like following



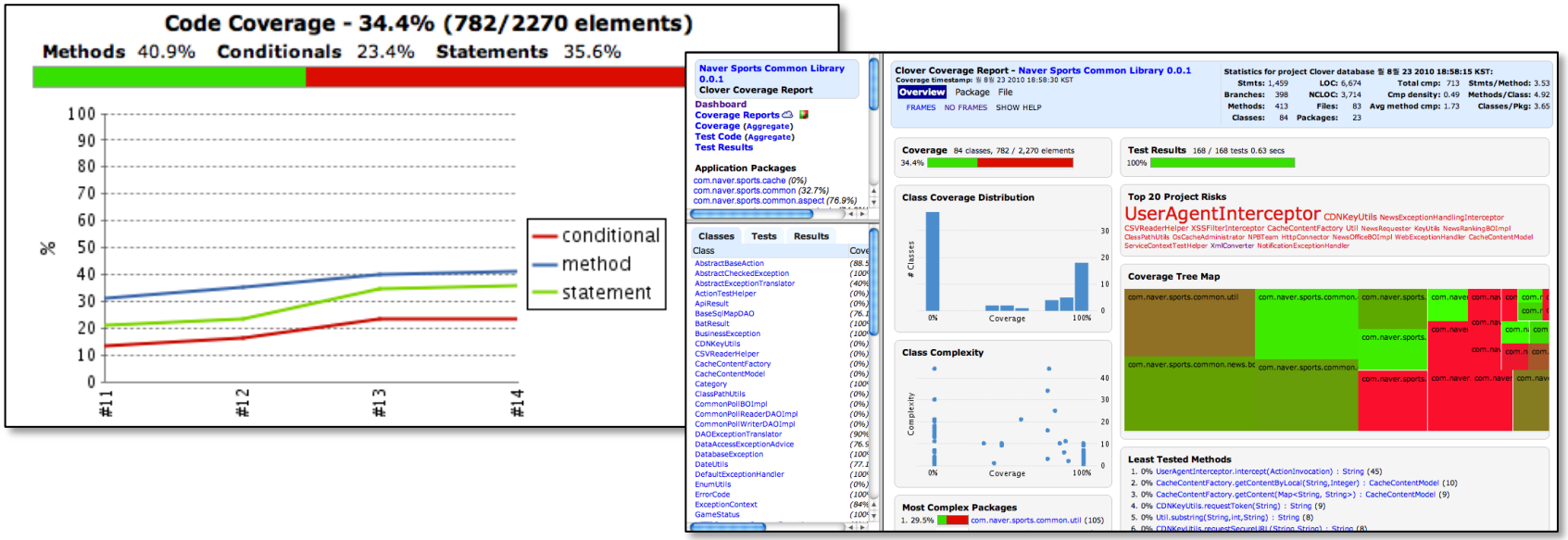
Add clover goal and enable clover reports

4.5 Code Coverage with Clover

Hudson Clover Plugin

Configure	Description
Clover report directory	Specify Clover Report(xml) location /target/site/clover in default
Clover report file name	Specify XML report file name clover.xml in default

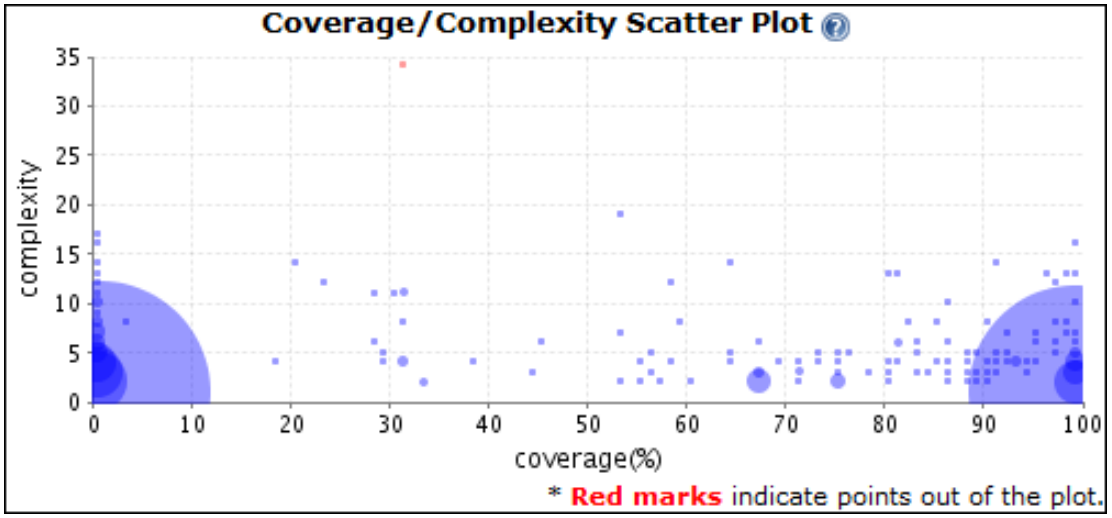
- After configuration, run "Build Now"
  - ✓ More than 2 builds with Clover, you'll see following



4.6 Coverage / Complexity Graph with CovComplPlot

Overview

- Hudson plugin which shows Coverage / Complexity relation graph which help the developer to choose the test necessary code.
- more complexity, more test!!



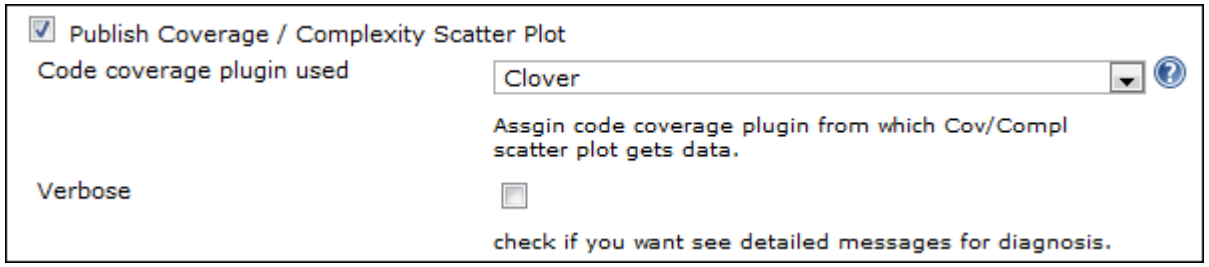
4.6 Coverage / Complexity Graph with CovComplPlot

Install and setup Hudson CovComplPlot Plugin

- Installable from Official Jenkins Update Site

❖ **How to setup CovComplPlot Plugin**

- Enable "Publish Coverage / complexity Scatter Plot" and choose coverage report type

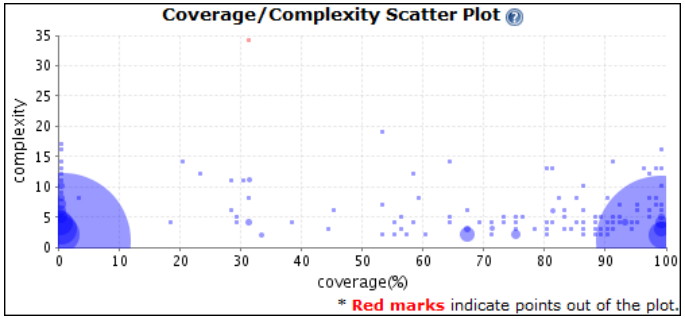


- Click "Build Now"

4.6 Coverage / Complexity Graph with CovComplPlot

Result

❖ **Graph**



❖ **Click each grid to see what methods are located in the each grid.**

Coverage / Complexity Scatter Plot

Method list(2055) in the range of coverage (90%~100%) and complexity (0~4)

Method	Complexity	Coverage(%)	Size	Covered
getPercentileValue(List<Double>,Double) : Double	4	94.74	19	18
getStatusCountList(List<JiraIssue>,boolean) : List<JiraStatistic>	4	93.75	16	15
getStrategyResultCountList(List<JiraIssue>,boolean) : List<JiraStatistic>	4	100	16	16
setAllBtsComponent(List<BtsComponent>) : void	4	93.33	15	14
setAllBtsIssueType(List<BtsIssueType>) : void	4	93.33	15	14
setAllBtsStatus(List<BtsStatus>) : void	4	93.33	15	14
setAllBtsVersion(List<BtsVersion>) : void	4	93.33	15	14
setAllProjectMember(List<ProjectMember>) : void	4	93.33	15	14
setAllCodeQualityGrade(List<CodeQualityGrade>) : void	4	93.33	15	14
getMemberInfo() : Map<String, String>	4	92.31	13	12
updateProjectQualityMeasure() : String	4	100	12	12
createSysLog(Class<?>,String,Object) : SysLog	4	90.91	11	10
deleteProjectMember() : String	4	100	11	11
updateProjectMilestone() : String	4	100	11	11
deleteProjectQualityMeasure() : String	4	90	10	9
insertProjectMember() : String	4	100	9	9
deleteProjectMilestone() : String	4	100	9	9
insertProjectMemberMulti(List<ProjectMember>) : int	4	100	9	9
updateProjectComment() : String	4	100	9	9
deleteProjectComment() : String	4	100	9	9

Enable CovComplPlot plugin

4.7 LOC & Complexity with N'SIQ Collector

N'SIQ Collector

❖ **General tool to measure code size and complexity**

❖ **What should Measure?**

- LOC : Code without comments and blanks
- Complexity : Depends on each team decision

❖ **Measures excludes followings**

- Patch or Code developed by other teams, open source, outsourcing…
- Auto generated code from Lex or Yacc.
- Windows message loop which have higher complexity in nature

4.7 LOC & Complexity with N'SIQ Collector

Install and setup N'SIQ Collector

❖ Install N'SIQ Collector

- Download N'SIQCollector binary (http://dev.naver.com/projects/nsiqcollector)
- Unzip the downloaded binary

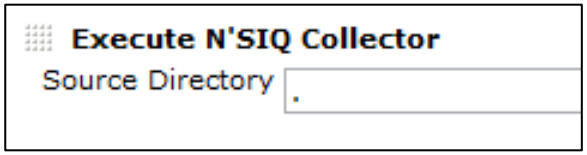
❖ Install Hudson N'SIQ Collector Plugin

- Installable from Official Jenkins Update Site
- Manage Hudson ➔ Configure System ➔ Configure N'SIQ Collector
  - ✓ Input N'SIQ Collector executable location

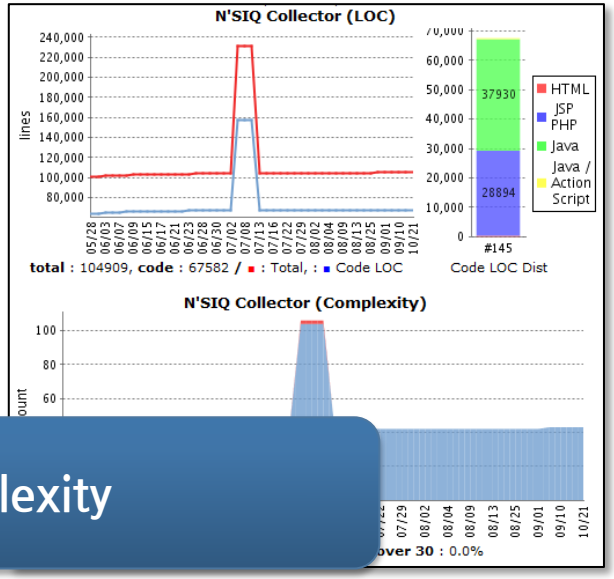
4.7 LOC & Complexity with N'SIQ Collector

Install and setup N'SIQ Collector

- **Setup N'SIQ Collector per Hudson plugin**
  - Configure ➔ Add build step ➔ Execute N'SIQ Collector
  - Put relative path to be analyzed in Source Directory field



  - Click "Publish N'SIQ Collector" in Post-build Actions
    - ✓ Enable all checkbox in sub menu
  - Save
- **Result**
  - Click "Build Now"



Add N'SIQ Collector LOC and Complexity

4.8 Static Analysis with Klocwork

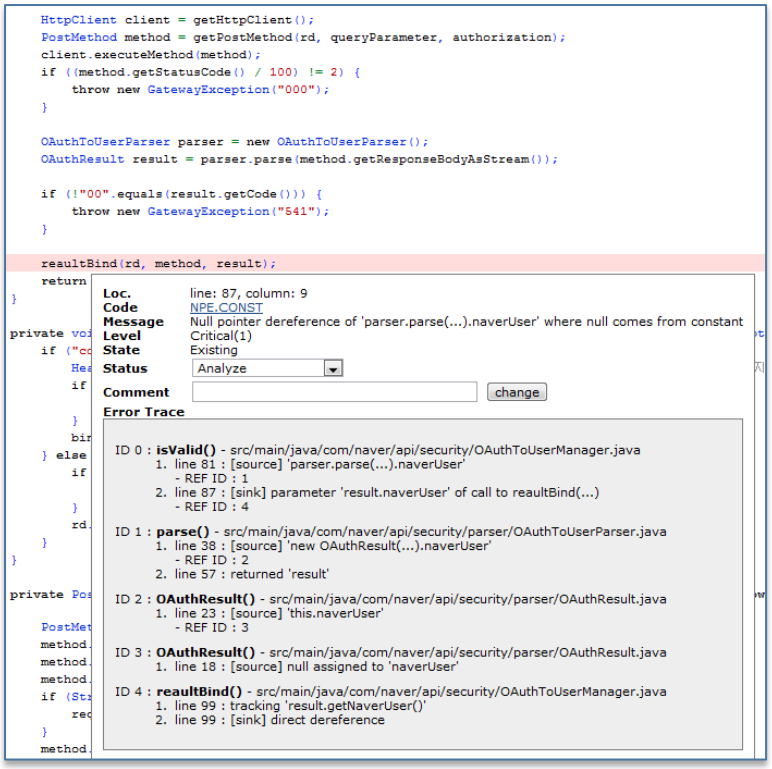
Static Analysis

Only Demo

- **Test vs Static Analysis**
  - Test = Test Case Execution Time + Defect Cause Analysis Time
  - Static Analysis = Only Analysis Time

- **Static analysis detects possible defects in build time like following**

- **Static analysis reports the step how the defect is reproduced.**



4.8 Static Analysis with Klocwork

Klocwork Rules

Only Demo

L1~L4 : 101

01(Critical)	**19**
Cross-site Scripting (XSS)	2
Data Injection	2
Denial of Service	1
Information Leaks	1
Possible Runtime Failures	6
Process and Path Injection	4
Suspicious Code Practices	1
Unvalidated User Input	2
02(Severe)	**40**
Android Issues	4
Denial of Service	3
Process and Path Injection	2
Redundant Code	5
Suspicious Code Practices	9
Threads and Synchronization Issues	2
Unvalidated User Input	1
Use After Free	11
Weak Encryption	3

03(Error)	**29**
Android Issues	4
Data Injection	2
Denial of Service	1
Ignored Return Values	3
Information Leaks	1
Possible Runtime Failures	1
Resource Leaks	15
Unsafe Code Practies	1
Unvalidated User Input	1
04(Unexpected)	**13**
Data Injection	1
Poor Error Handing	1
Possible Runtime Failures	1
Redundant Code	2
Suspicious Code Practices	3
Threads and Synchronization Issues	4
Weak Encapsulation	1

4.8 Static Analysis with Klocwork

How static analysis works

Only Demo

- **NPE.COND Defect**
  - Guess if static analysis assumes that all method parameter can be given as null value

    ➔ Excessive false alarm

  - What if there is a condition in which check the parameter is null or not…

    ➔ Is it safe to say null value can be given as parameter?

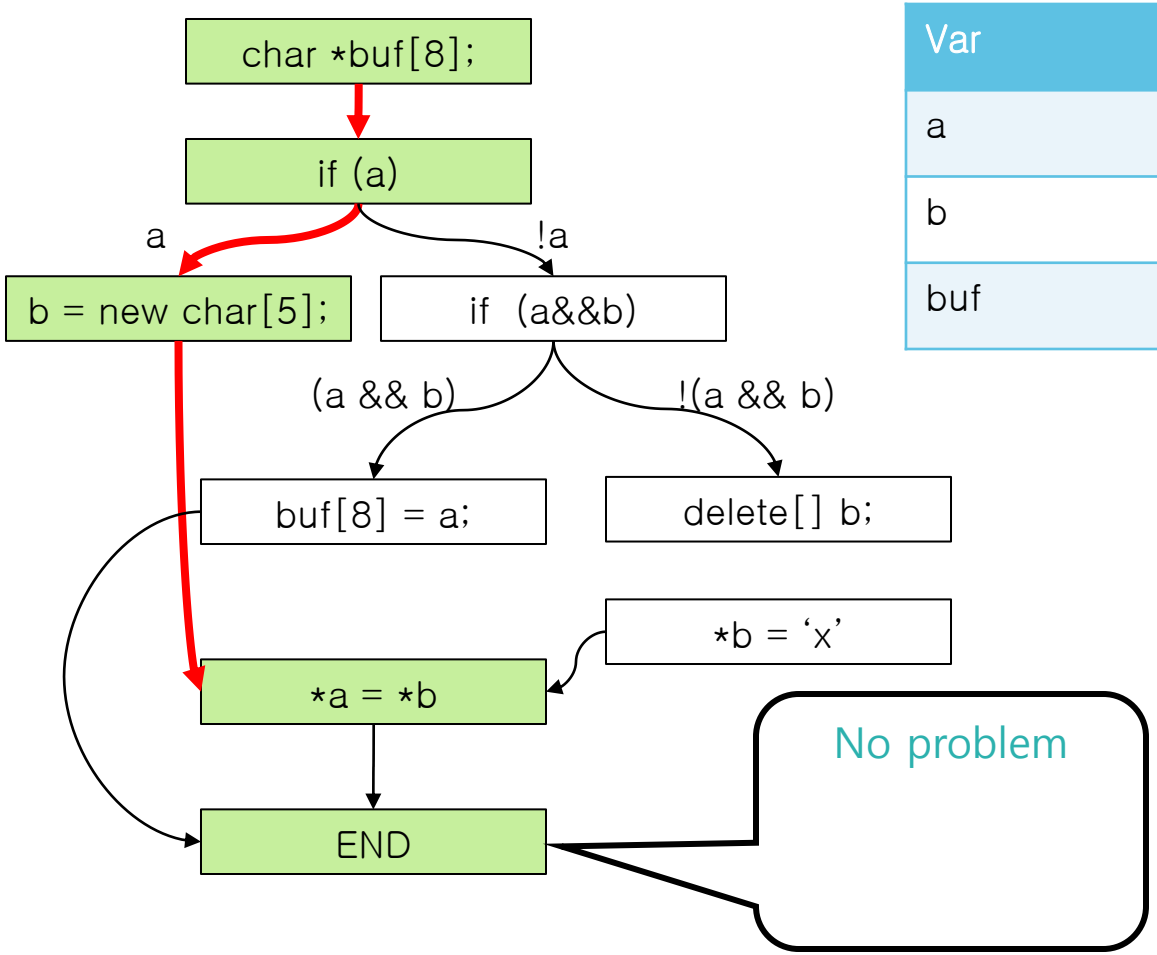
```
#define SIZE 8
void set(char *a, char *b){
  char *buf[SIZE];
  if (a) {
    b = new char[5];
  } else {
    if (a && b) {
      buf[size] = a;
      return;
    } else {
      delete[] b;
    }
    *b = 'x';
  }
  *a = *b;
}
```

4.8 Static Analysis with Klocwork

How static analysis works

Only Demo

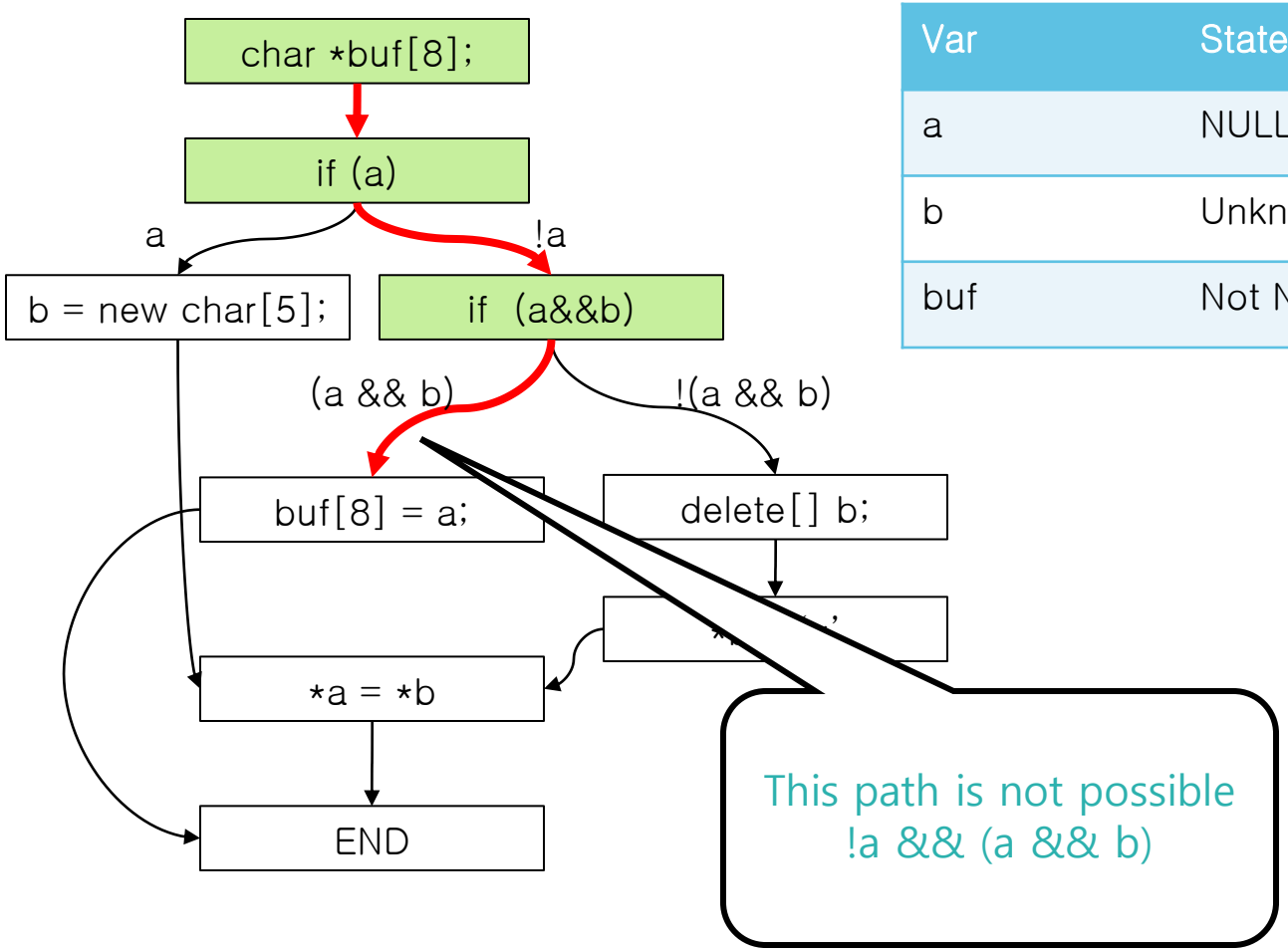
❖ Value Tracing



Var	State
a	Not NULL
b	Not NULL
buf	Not NULL

Only Demo

❖ Impossible path



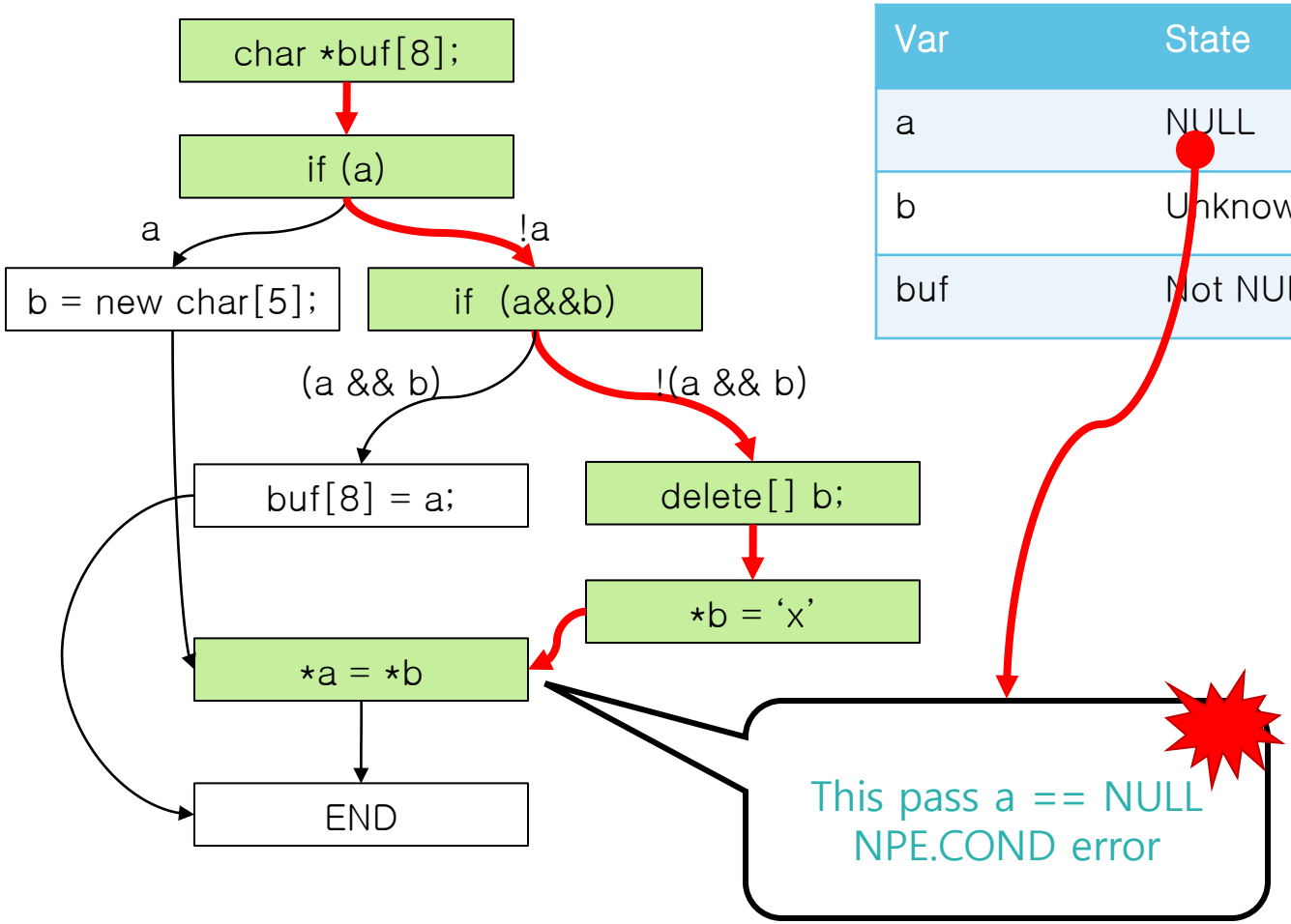
Var	State
a	NULL
b	Unknown
buf	Not NULL

4.8 Static Analysis with Klocwork

How static analysis works

Only Demo

❖ Error Case



Var	State
a	NULL
b	Unknown
buf	Not NULL

4.8 Static Analysis with Klocwork

Limitation

Only Demo

❖ Limitation on Static Analysis

- If there is no Source Code
  - ✓ Lib / Dll / Jar
- If the value is given from external env
  - ✓ socket / scanf
- If no one know which class will be wired in the other class in compile time
  - ✓ E.g) Spring

❖ How to overcome.

- Remove unnecessary interfaces.
- Provide Knowledge base
  (a user defined info about the methods with no source code)

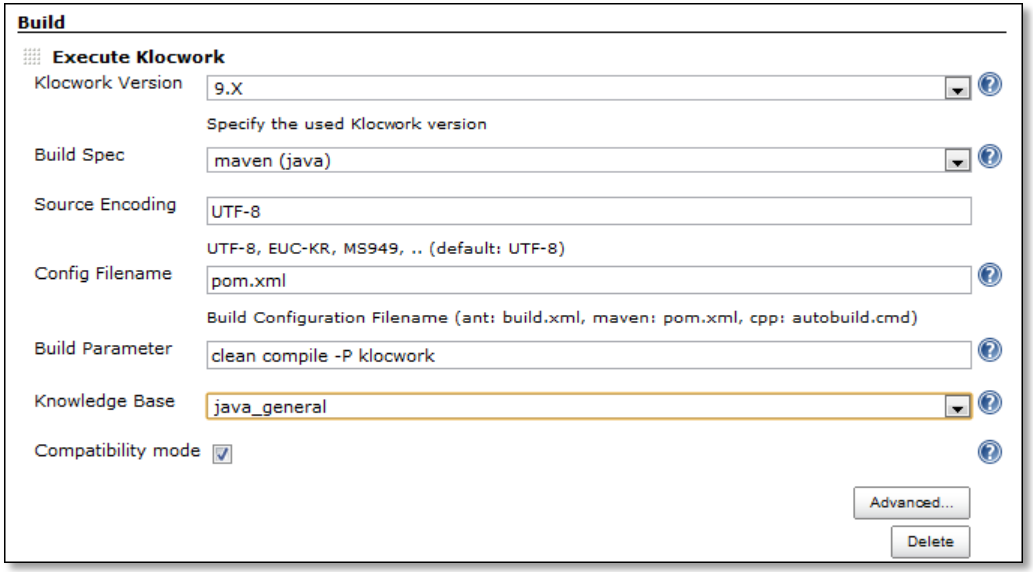
4.8 Static Analysis with Klocwork

Configure Klocwork Plugin per project

Only Demo

❖ Add "Klocwork Builder"

- Configure ➔ Add build step ➔ Click "Execute Klocwork"
- Input "Execute Klocwork" configuration.
  - ✓ Build Spec
    - "maven"
  - ✓ Config Filename
    - "pom.xml"
  - ✓ Build Parameter
    - "kw:run –P klocwork"
  - ✓ Knowledge Base
    - Additional Info for library
    - "Java_general"



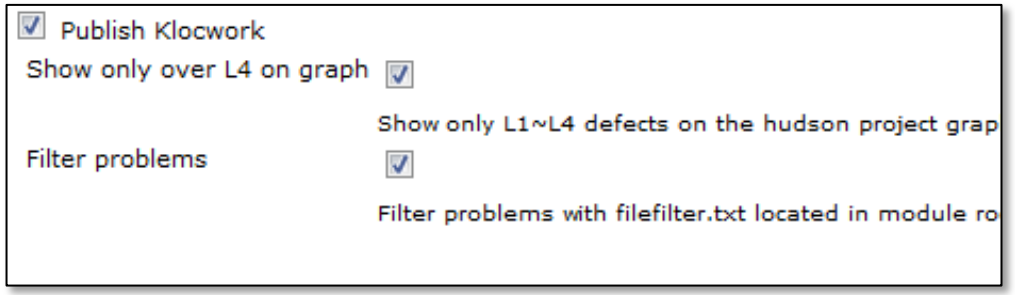
4.8 Static Analysis with Klocwork

Configure Klocwork Plugin per project

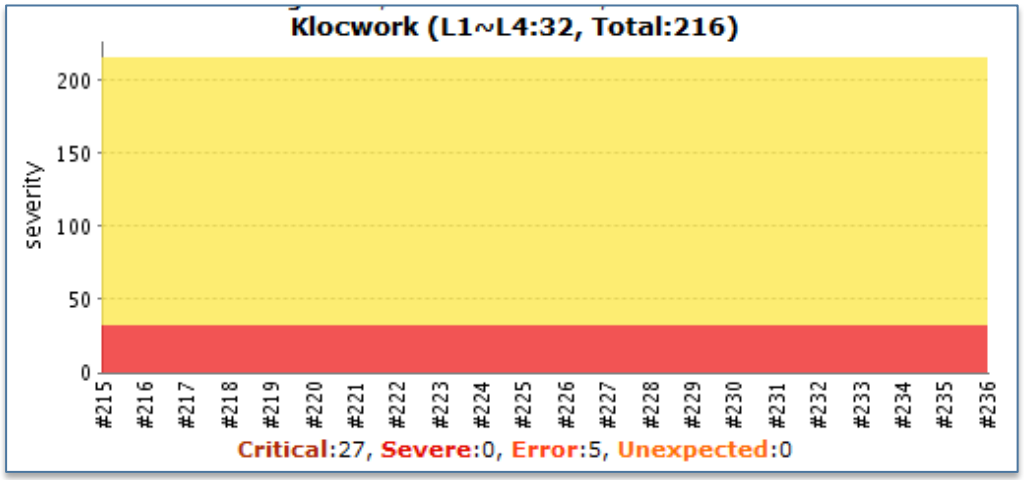
Only Demo

❖ Enable Klocwork Publisher

- Click "Publish Klocwork" in Post-build Action
- Show only over L4 on graph : Click If you want to see only L1~L4 errors



❖ Click Build Now



4.8 Static Analysis with Klocwork

Configure Klocwork Plugin per project

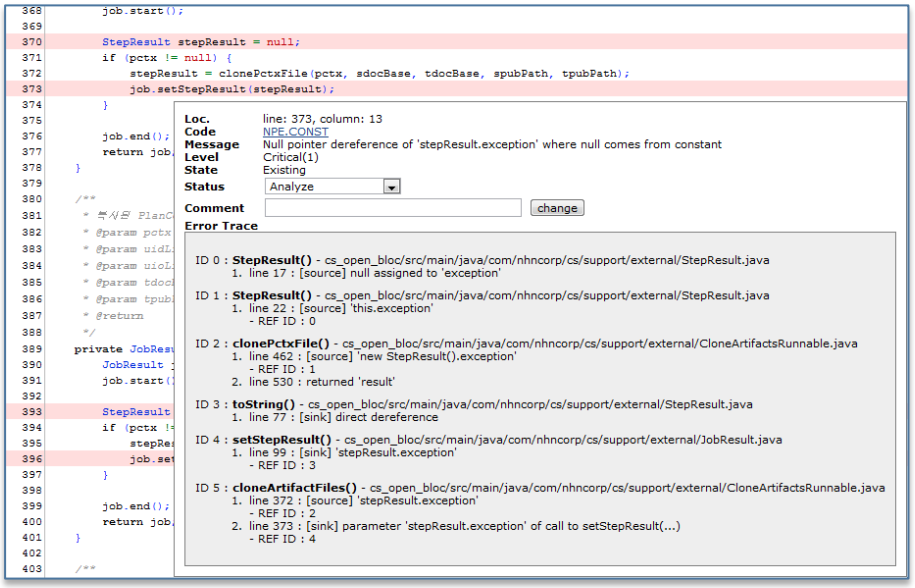
Only Demo

- **Click the shown graph.**
  - Dig into the defect details

Klocwork Result

Summary

item	crit.	severe	error	unexp.	i
cs_open_bloc/src/main/java/com/nhncorp/cs/bo	4	0	0	0	
cs_open_bloc/src/main/java/com/nhncorp/cs/common	0	0	0	0	
cs_open_bloc/src/main/java/com/nhncorp/cs/dao	0	0	0	0	
cs_open_bloc/src/main/java/com/nhncorp/cs/dto	0	0	0	0	
cs_open_bloc/src/main/java/com/nhncorp/cs/support/ds	1	0	4	0	
cs_open_bloc/src/main/java/com/nhncorp/cs/support/encryption	5	0	0	0	
cs_open_bloc/src/main/java/com/nhncorp/cs/support/external	12	0	0	0	
cs_open_bloc/src/main/java/com/nhncorp/cs/support/login	1	0	0	0	
cs_open_bloc/src/main/java/com/nhncorp/cs/util	4	0	1	0	



Add “Execute Klocwork” build step
Enable “Publish Klocowork”
Click “Build Now” and see defects found

4.9 Coding Standard Conformance with Checkstyle

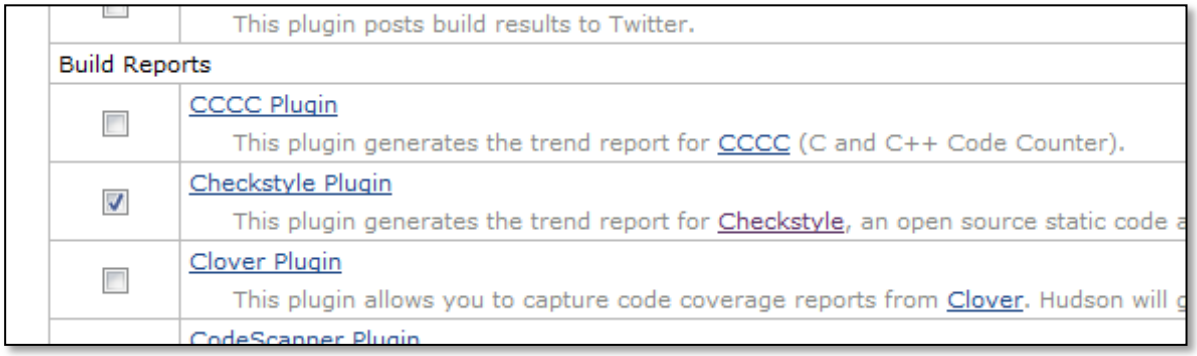
Checkstyle

- **Most famous Java coding style checker**
  - NHN defined our own coding style rules by customizing checkstyle rules.

4.9 Coding Standard Conformance with Checkstyle

Install Hudson Checkstyle Plugin

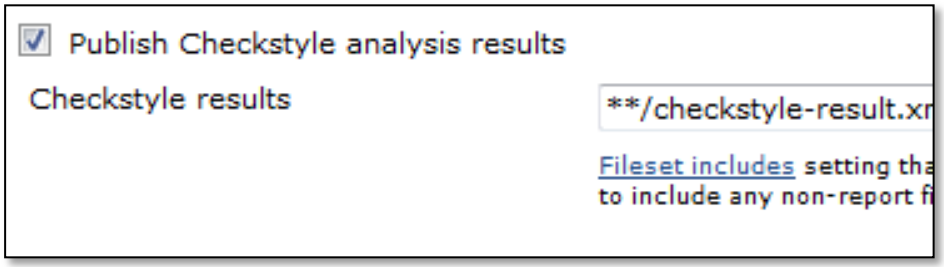
- Configure Hudson ➔ Manage plugins ➔ Available Tab ➔ Click Checkstyle Plugin ➔ Install ➔ restart

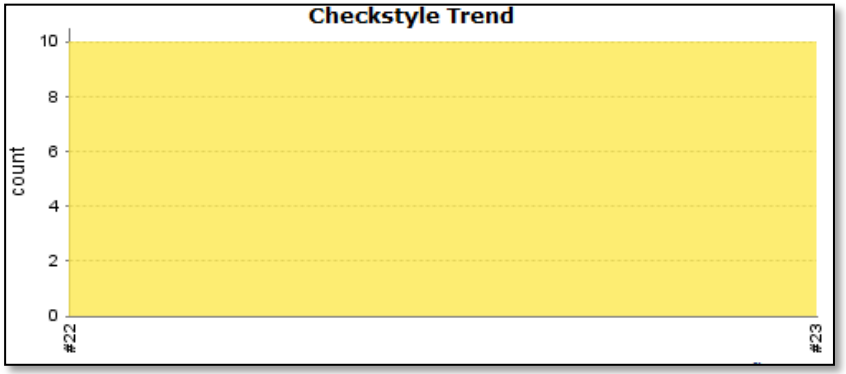


4.9 Coding Standard Conformance with Checkstyle

Configure Hudson Checkstyle Plugin per project

- **Add checkstyle goal in maven goal list**
  - Add checkstyle:checkstyle goal in front of other goals
- **Add checkstyle publisher**
  - Check "Publish Checkstyle analysis results" and specify the Checkstyle xml results path





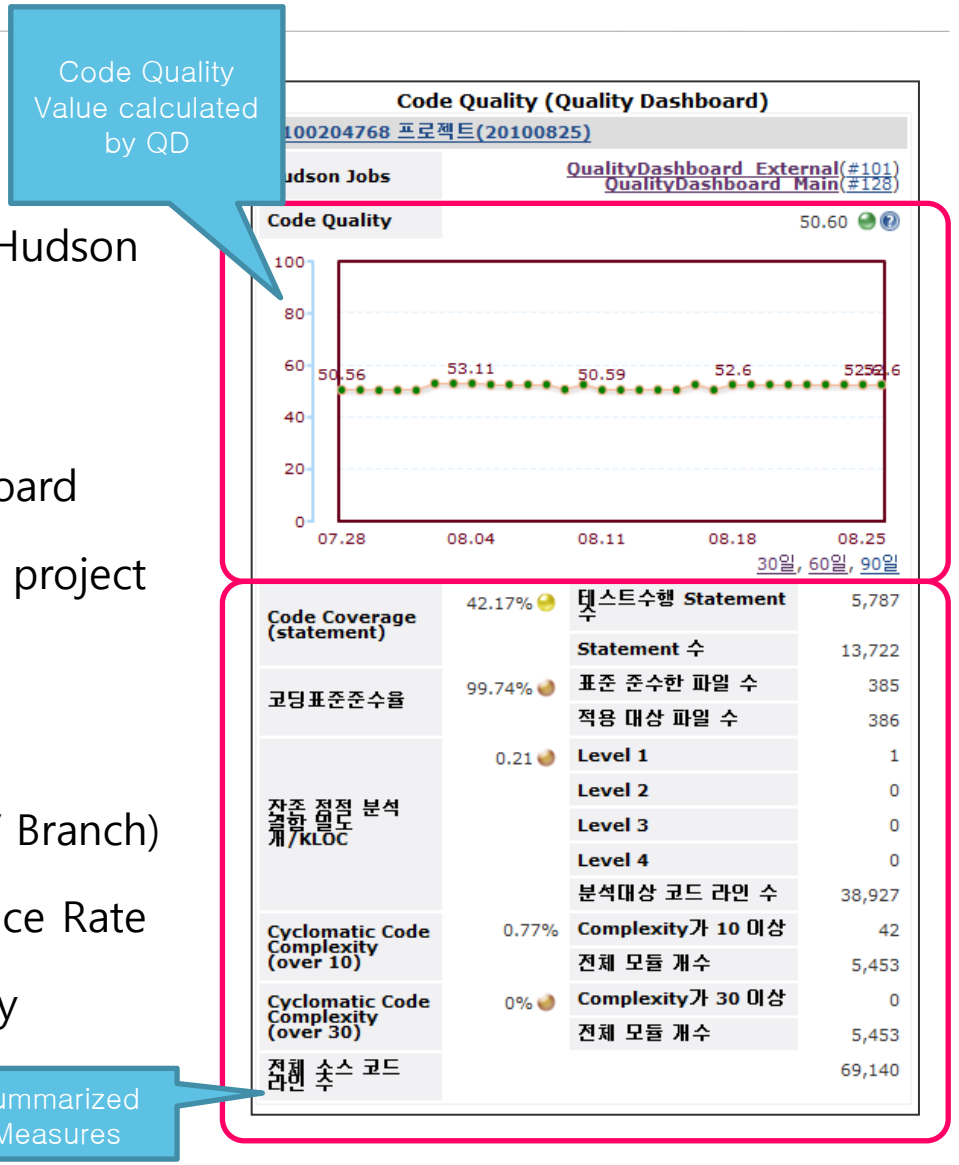
- **Run "Build Now"**
  - Dig into graph

Confiugure Checkstyle in Hudson project
Click "Build Now" and see found violations

4.9 Calculate QP metrics with QD plugin

Quality Dashboard Plugin

- **Hudson QD PlugIn**
  - Send collected metric from Hudson to Quality Dashboard
  - Show Code Quality Value calculated by Quality Dashboard
  - Summarize multiple Hudson project metrics
- **Measures**
  - Code Coverage (Statement / Branch)
  - Coding Standard Conformance Rate
  - Static Analysis Defect Density
  - Complexity / LOC

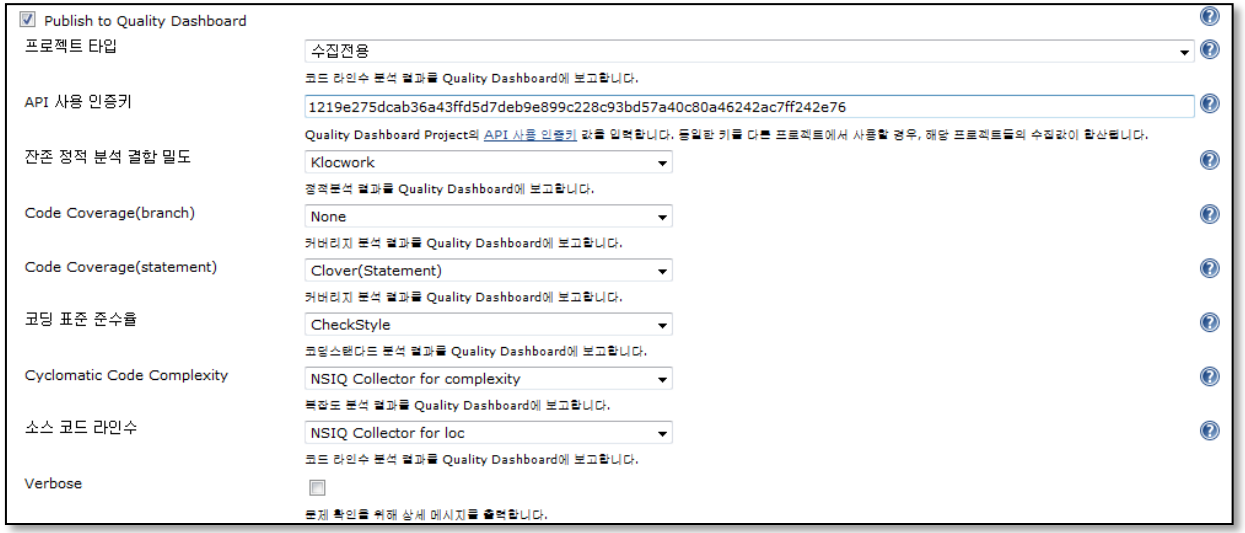


4.9 Calculate QP metrics with QD plugin

Configure Quality Dashboard Plugin

❖ Configure QD Plugin in a project

- Configure Project ➔ Enable "Publish to Quality Dashboard"
  - ✓ Select the Hudson Plugins used.



4.9 Calculate QP metrics with QD plugin

Project Type

❖ values

- Collect/Send : When you want to send the metrics if the metrics are collected.
- OnlyCollect : When you want to only collect metrics.
- OnlySend : When you want to send metrics collected by the other projects
- NoCollect/NoSend : When you disable this project

❖ How to use

✓ When you like to summarize A, B, C project and You want to send the collected metrics only when C is built.
- A : OnlyCollect, B : OnlyCollect, C: Collect/Send
- A, B, C's API Key should be same

✓ When you like to summarize A, B, C project whenever each project is built, However you want to send the metrics to QualityDashboard one a week.
- A : OnlyCollect, B : OnlyCollect, C: OnlyCollect,
  Create Separate Dummy D Project and set it OnlySend
- A, B, C, D's API Key should be same

Enable the collection of Coverage / Coding Style / Cyclomatic Complexity / LOC / Static Analysis Defect Density

Thanks