

소프트웨어 V&V

CTIP - Advanced

201511243 김동언

201511246 김상재

201511262 박우진

201711356 천세진

Index

1. CTIP Advanced 구축 Overview

1. Jacoco

1. Static Analysis

a. CheckStyle

b. PMD

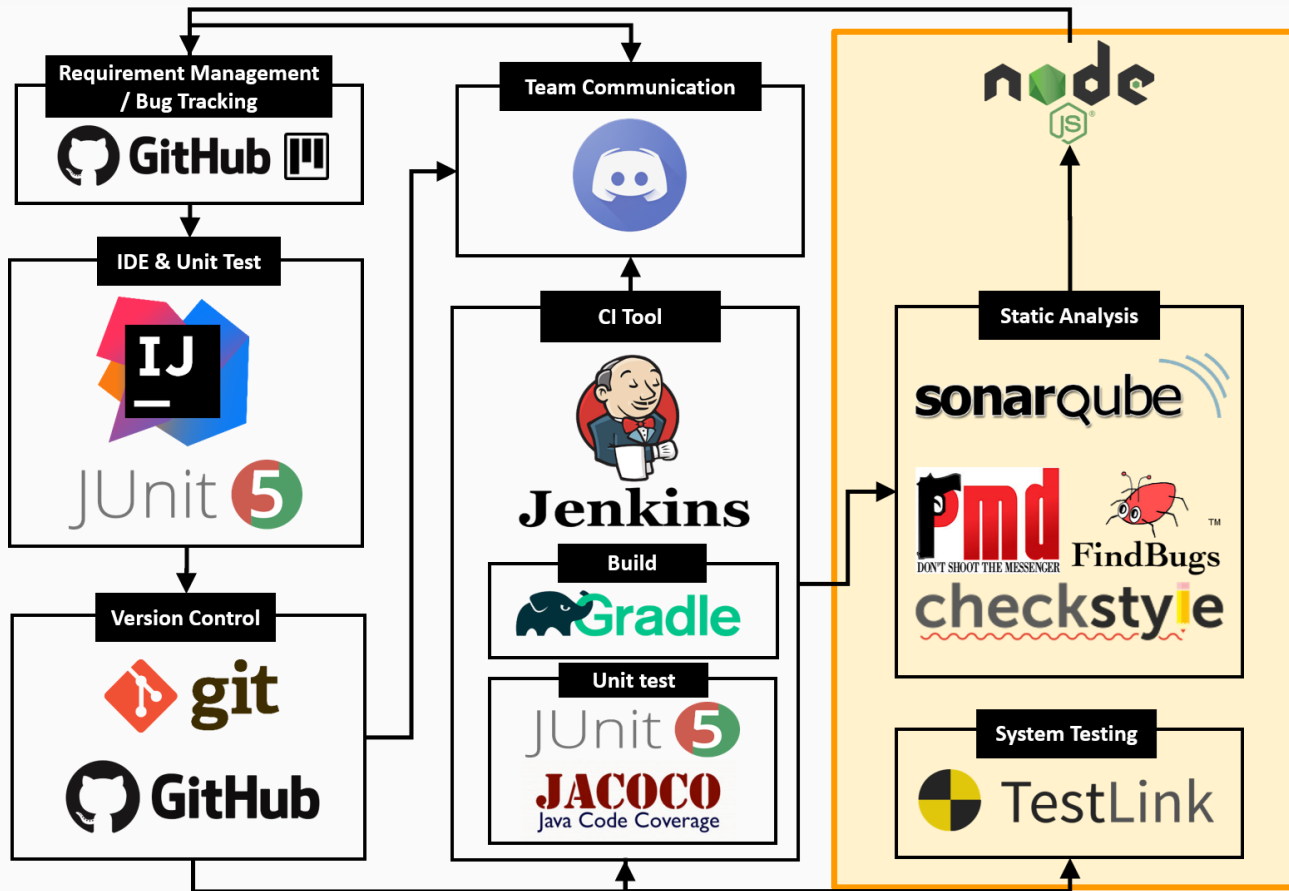
c. FindBugs

d. Sonarqube

1. Testlink

1. Overall CTIP

CTIP Advanced 구축 Overview



● Jacoco

- JVM 기반 환경에서 Code Coverage analysis 을 위한 도구.
- 객체지향방법론팀에서 만든 Unit Testing Code 가 얼마나 Code를 Cover 하는지 검사
- Code Coverage Criteria
 - 코드 커버리지에 대한 국제적으로 통용되는 기준 수치는 없음.
 - 100%는 현실적으로 거의 불가능하며 80% 정도를 목표로함.
 - 프로젝트별로 최소 60% 이상 커버리지를 목표로 하되, 중요 모듈의 경우 80%까지 상향 조정을 권고함.





Conditions

[Add Condition](#)

Conditions on New Code

Metric	Operator	Value	Edit	Delete
Condition Coverage	is less than	60.0%		

Conditions on Overall Code

Metric	Operator	Value	Edit	Delete
Line Coverage	is less than	60.0%		
Unit Test Success (%)	is less than	100%		

Coverage [Measures](#)



2
Unit Tests 

77.8%
Coverage on
9 New Lines to Cover

Static Analysis


Checkstyle, PMD, Findbugs



Static Analysis - Checkstyle

● Checkstyle

- Java 소스코드의 기술 형식 코딩규약을 지키는 지 체크하는 정적분석 툴
- 정해진 코딩규약에 위반되는 것들을 검사
- 직접 코딩 규약을 만들어 사용 가능
- 총 규칙 수 132 개

Checkstyle is a development tool to help programmers write Java code that adheres to a coding standard. It automates the process of checking Java code to spare humans of this boring (but important) task. This makes it ideal for projects that want to enforce a coding standard.

Checkstyle is highly configurable and can be made to support almost any coding standard. An example configuration files are supplied supporting the [Sun Code Conventions](#) , [Google Java Style](#).

A good example of a report that can be produced using Checkstyle and Maven  can be [seen here](#) .

About

Checkstyle

- Release Notes
- Consulting
- Sponsoring

Documentation

- ▼ Configuration
 - Property Types
 - Filters
 - File Filters
- ▼ Running
 - Ant Task
 - Command Line
- ▼ Checks
 - Annotations
 - Block Checks
 - Class Design
 - Coding
 - Headers
 - Imports
 - Javadoc Comments
 - Metrics
 - Miscellaneous
 - Modifiers
 - Naming Conventions
 - Regexp
 - Size Violations
 - Whitespace
- ▼ Style Configurations
 - Google's Style
 - Sun's Style

Static Analysis - PMD

● PMD - Programming Mistake Detector

- 정의된 규칙을 기반으로 소스 코드를 검사하여, 오류 및 위험 요인을 식별
- 사용되지 않았거나 최적화되지 않은 코드들 검색
- 총 규칙수 234 개

Overview

PMD is a static source code analyzer. It finds common programming flaws like unused variables, empty catch blocks, unnecessary object creation, and so forth. It's mainly concerned with **Java and Apex**, but **supports six other languages**.

PMD features many **built-in checks** (in PMD lingo, *rules*), which are documented for each language in our Rule references. We also support an extensive API to **write your own rules**, which you can do either in Java or as a self-contained XPath query.

PMD is most useful when integrated into your build process. It can then be used as a quality gate, to enforce a coding standard for your codebase. Among other things, PMD can be run:

PMD 6.23.0

About	▼
User Documentation	▼
Rule Reference	▲
Apex Rules	▼
EcmaScript Rules	▼
Java Rules	▲
Index	
Best Practices	
Code Style	
Design	
Documentation	
Error Prone	
Multithreading	
Performance	
Security	

Static Analysis - FindBugs

● FindBugs

- 작성한 프로그램의 잠재적인 결함을 찾아주는 도구
- 컴파일된 클래스 파일에서 바이트 코드를 읽어서 사용
- 총 규칙 수 408개

FindBugs is an open-source static code analyser created by [Bill Pugh](#) and David Hovemeyer which detects possible bugs in [Java](#) programs.^{[2][3]} Potential errors are classified in four ranks: (i) scariest, (ii) scary, (iii) troubling and (iv) of concern. This is a hint to the developer about their possible impact or severity.^[4] FindBugs operates on [Java bytecode](#), rather than source code. The software is distributed as a stand-alone [GUI](#) application. There are also plug-ins available for [Eclipse](#),^[5] [NetBeans](#),^[6] [IntelliJ IDEA](#),^{[7][8][9]} [Gradle](#), [Hudson](#),^[10] [Maven](#),^[11] [Bamboo](#)^[12] and [Jenkins](#).^[13]

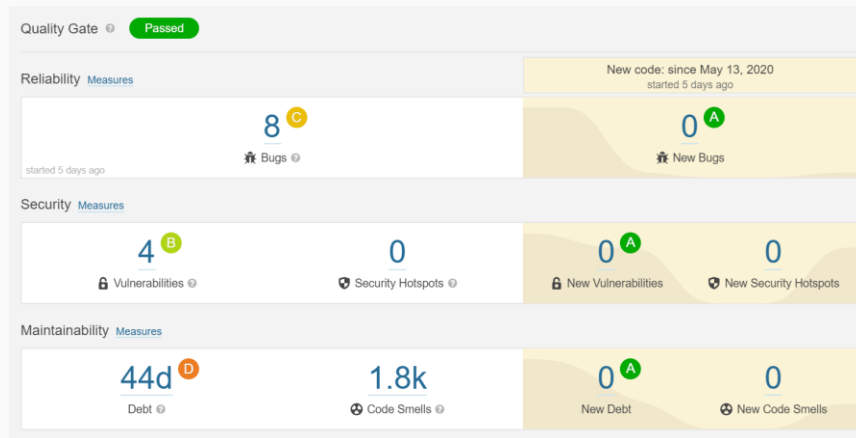
Additional rule sets can be plugged in FindBugs to increase the set of checks performed.^[14]

Categories	Count
Correctness	142
Bad Practice	84
Dodgy Code	71
Multithreaded Correctness	45
Performance	27
Malicious Code Vulnerability	15
Security	11
Experimental	3
Internationalization	2

Static Analysis - Sonarqube

● Sonarqube

- 정적 코드 분석 수행 플랫폼
- 프로그래밍 언어에서 Bug, Code Smell, Vulnerability 을 발견함을 목적으로 함
- 의존성, 보안성, 유지보수, 커버리지, 중복 코드에 대한 분석 수행
- CI Tool (Jenkins)과 연동 가능하며, 정적 분석 검사 도구들 (PMD, CheckStyle, FindBugs)과 연동
- Sonarsource Rules
 - <https://rules.sonarsource.com/>
 - Sonarqube 자체적으로 분석 rules 또한 존재



Static Analysis - Sonarqube

The screenshot displays the Sonarqube interface for configuring rules. The top navigation bar includes 'sonarqube', 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', and 'Administration'. A search bar is present with the text 'Search for projects and files...'. Below the navigation, there are controls for 'Bulk Change' and 'Clear All Filters'. The main content area is a list of rules, each with a status icon, a description, associated tags, and a 'Deactivate' button.

Rule ID	Description	Language	Category	Tags	Action
S1118	"equals()" should not be used to test the values of "Atomic" classes	Java	Bug	multi-threading	Deactivate
S1119	"==" should not be used instead of "=="	Java	Bug		Deactivate
S1120	"==" and "!=" should not be used when "equals" is overridden	Java	Code Smell	cert, cwe, suspicious	Deactivate
S1121	"@CheckForNull" or "@Nullable" should not be used on primitive types	Java	Code Smell		Deactivate
S1122	"@Controller" classes that use "@SessionAttributes" must call "setComplete" on their "SessionStatus" objects	Java	Bug	spring	Deactivate
S1123	"@Deprecated" code marked for removal should never be used	Java	Code Smell	cert, cwe, obsolete	Deactivate
S1124	"@Deprecated" code should not be used	Java	Code Smell	cert, cwe, obsolete	Deactivate
S1125	"@EnableAutoConfiguration" should be fine-tuned	Java	Code Smell	performance, spring	Deactivate
S1126	"@Import"s should be preferred to "@ComponentScan"s	Java	Code Smell	performance, spring	Deactivate
S1127	"@NonNull" values should not be set to null	Java	Bug	cert, cwe	Deactivate
S1128	"@Override" should be used on overriding and implementing methods	Java	Code Smell	bad-practice	Deactivate
S1129	"@RequestMapping" methods should be "public"	Java	Vulnerability	owasp-a6, spring	Deactivate
S1130	"@RequestMapping" methods should specify HTTP method	Java	Vulnerability	cwe, owasp-a6, sans-top25-insecure, ...	Deactivate
S1131	"@SpringBootApplication" and "@ComponentScan" should not be used in the default package	Java	Bug	spring	Deactivate
S1132	"action" mappings should not have too many "forward" entries	Java	Code Smell	brain-overload, struts	Deactivate
S1133	"ActiveMQConnectionFactory" should not be vulnerable to malicious code deserialization	Java	Vulnerability	cwe, owasp-a8	Deactivate
S1134	"Arrays.stream" should be used for primitive arrays	Java	Code Smell	performance	Deactivate

The left sidebar contains the following filters:

- Language: Search for rules...
- Type:
 - Bug: 586
 - Vulnerability: 164
 - Code Smell: 814
 - Security Hotspot: 34
- Tag
- Repository:
 - SonarAnalyzer Java: 515
 - FindBugs Java: 448
 - FindBugs Contrib Java: 306
 - Find Security Bugs Java: 115
 - Checkstyle Java: 114
 - PMD Java: 80
 - PMD Unit Tests Java: 15
 - Common Java Java: 5
- Default Severity
- Status
- Security Category
- Available Since
- Template
- Quality Profile: SVV_PR... (Clear)
- Inheritance
- Activation Severity

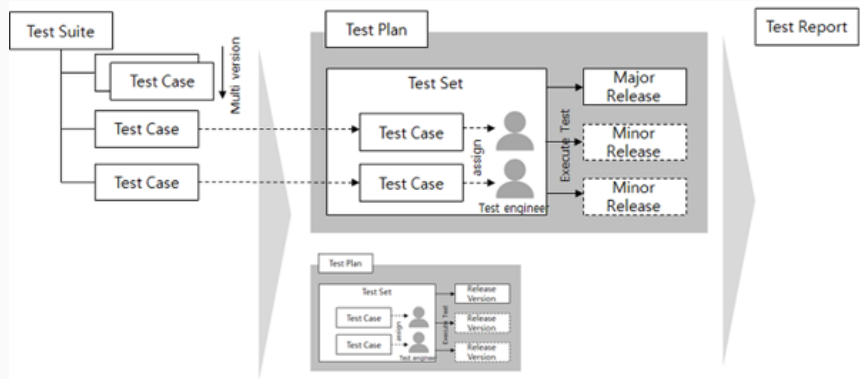
System Testing

Testlink

System Testing - TestLink

● TestLink

- 전체 시스템에 대해 어떻게 테스트를 하고, 테스트에 대한 내용을 관리하는 도구
- 테스트 케이스를 여러 사람이 나눠서 테스트를 진행하고 결과를 기입함



Test Results on Build BUILD-52

Print Show complete execution history Import XML Results

Execute and Save Results

Test Plan TestPlan_0517

Build BUILD-52

Test Suite : BruteForceTesting/

Last execution (any build)

Date : 05/18/2020 11:13:02 - Tested by : svv5 - Build : BUILD-52 - Status : Passed

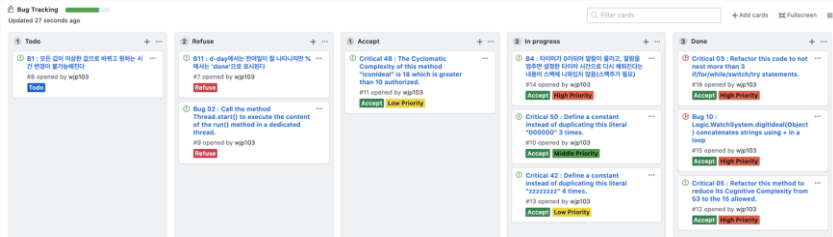
Last execution (current build) - Build : BUILD-52

Date	Build	Tested by	Status	Exec (min)	Version	Run mode
05/18/2020 11:13:02	BUILD-52	svv5	Passed	1		

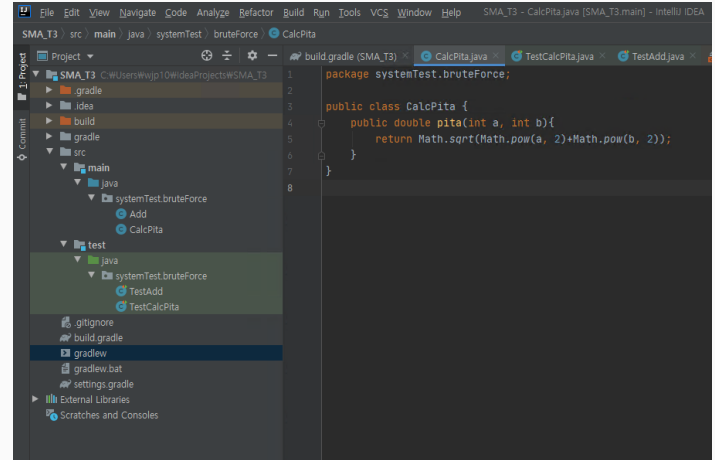
Overall CTIP

Overall CTIP

- Requirement Management / Bug Tracking (Github Project)

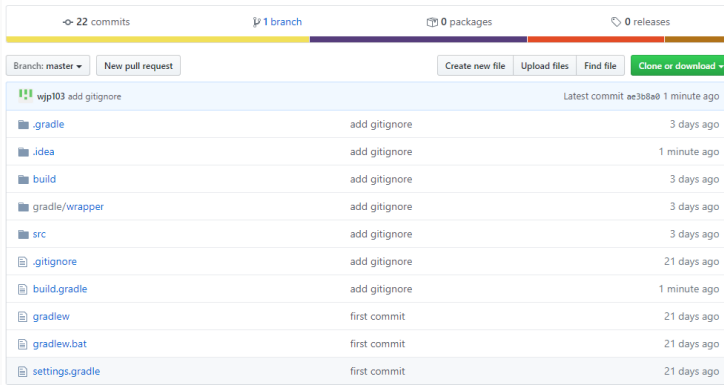


- IDE (IntelliJ)



Overall CTIP

● Code Configuration Management (Git)

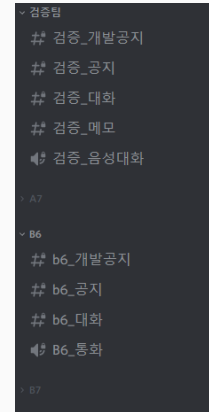


The screenshot shows a Git repository interface with the following details:

- 22 commits, 1 branch, 0 packages, 0 releases
- Branch: master
- Buttons: New pull request, Create new file, Upload files, Find file, Clone or download
- Commit: wjpt103 add gitignore (Latest commit ac368a0 1 minute ago)
- Files changed table:

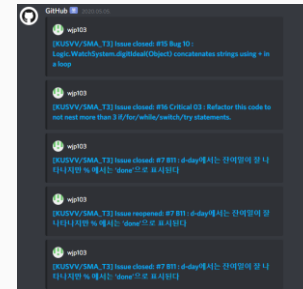
File	Change	Time
.gradle	add gitignore	3 days ago
.idea	add gitignore	1 minute ago
build	add gitignore	3 days ago
gradle/wrapper	add gitignore	3 days ago
src	add gitignore	3 days ago
.gitignore	add gitignore	21 days ago
build.gradle	add gitignore	1 minute ago
gradlew	first commit	21 days ago
gradlew.bat	first commit	21 days ago
settings.gradle	first commit	21 days ago

● Team Communication (Discord)



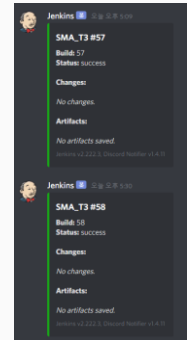
The screenshot shows a Discord chat window with the following content:

- 검증팀
- ## 검증_개발공지
- ## 검증_공지
- ## 검증_대화
- ## 검증_메모
- ## 검증_음성대화
- A7
- B6
 - ## b6_개발공지
 - ## b6_공지
 - ## b6_대화
 - ## b6_통화
- B7



The screenshot shows a GitHub pull request discussion in a Discord channel:

- GitHub
- git103: [XUSVV/SMA_T3] Issue closed: #10 Bug 10: `String toString()` method (Object) constructor string using + in a loop
- git103: [XUSVV/SMA_T3] Issue closed: #16 Critical Q1: Refactor this code to not use more than 30 'for' while/switch/try statements
- git103: [XUSVV/SMA_T3] Issue closed: #7 B11 1-day에서는: 잔여량이 못 나거나 시간 % 에서는 'done'으로 표시된다
- git103: [XUSVV/SMA_T3] Issue reopened: #7 B11 1-day에서는: 잔여량이 못 나거나 시간 % 에서는 'done'으로 표시된다
- git103: [XUSVV/SMA_T3] Issue closed: #7 B11 1-day에서는: 잔여량이 못 나거나 시간 % 에서는 'done'으로 표시된다



The screenshot shows two Jenkins build status notifications in a Discord channel:

- Jenkins SMA_T3 #57
 - Build 57
 - Status: success
 - Changes: No changes
 - Artifacts: No artifacts saved
- Jenkins SMA_T3 #58
 - Build 58
 - Status: success
 - Changes: No changes
 - Artifacts: No artifacts saved

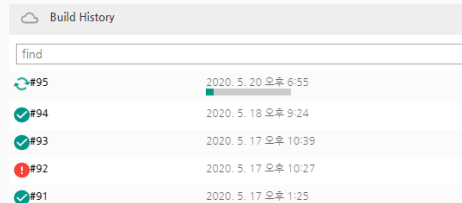
Overall CTIP

● CI Server (Jenkins)



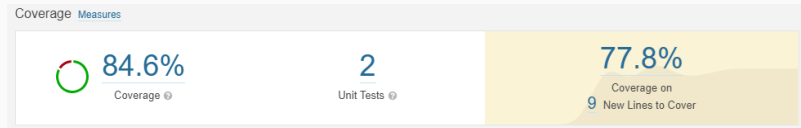
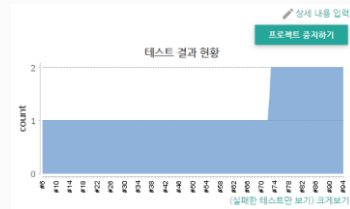
The screenshot shows the Jenkins interface for Project SMA_T3. On the left is a navigation menu with items like '상태', '변경사항', '작업공간', 'Build Now', 'Project 삭제', '구성', 'GitHub Hook Log', 'GitHub', 'SonarQube', and 'Rename'. The main area displays 'Project SMA_T3' with a '프로젝트 중지하기' button. Below this, there are sections for 'SonarQube' (with '작업공간' and '최근 변경사항' links) and 'SonarQube Quality Gate' (showing 'SMA_T3 OK' and 'server-side processing: Success'). A '고정링크' section lists build details: 'Last build. (#56). 40 min 전', 'Last stable build. (#56). 40 min 전', 'Last successful build. (#56). 40 min 전', and 'Last failed build. (#54). 57 min 전'. At the bottom left, a 'Build History' table shows builds #56 and #55.

● Automatic Build (Gradle) Unit Testing (JUnit) Code Coverage (Jacoco)



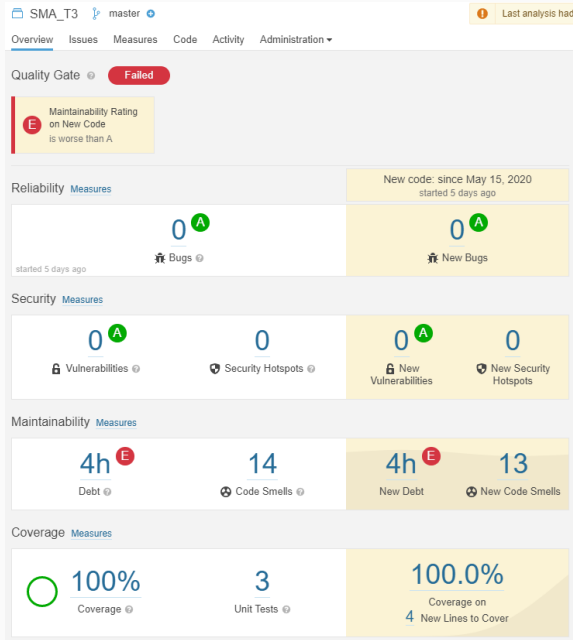
The screenshot shows the 'Build History' table with a search bar and a list of builds:

Build Number	Time
#95	2020. 5. 20 오후 6:55
#94	2020. 5. 18 오후 9:24
#93	2020. 5. 17 오후 10:39
#92	2020. 5. 17 오후 10:27
#91	2020. 5. 17 오후 1:25

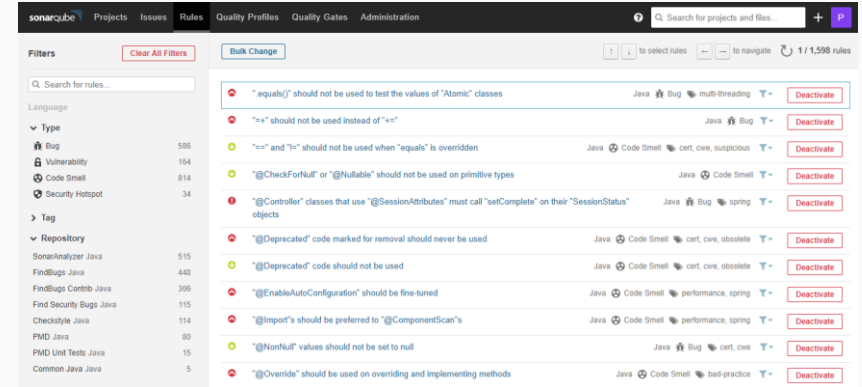


Overall CTIP

● Static Code Analysis (Sonarqube)

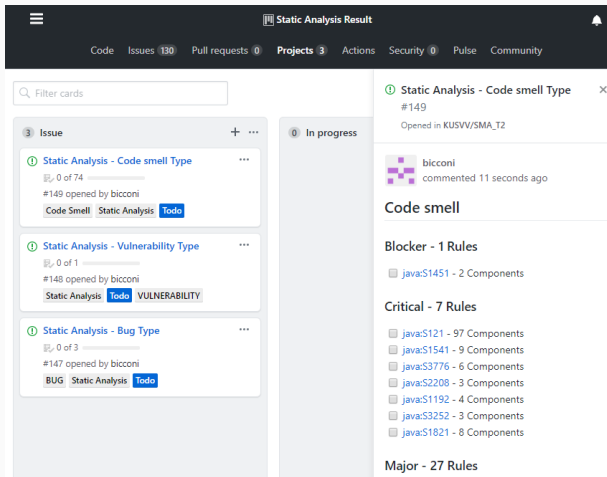


● PMD, Checkstyle, Findbugs



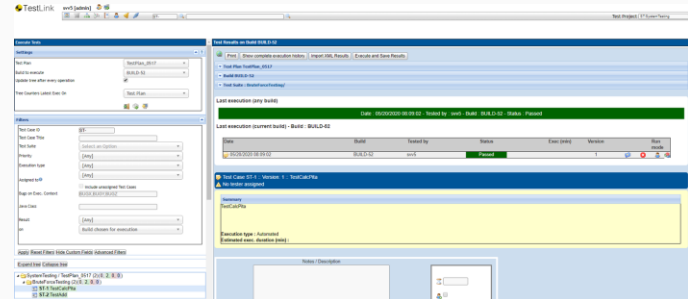
Overall CTIP

- Sonarqube Results
(making Github project issue)



The screenshot shows the Sonarqube interface for a static analysis result. The top navigation bar includes 'Code', 'Issues 139', 'Pull requests 0', 'Projects 3', 'Actions', 'Security 0', 'Pulse', and 'Community'. A search bar is present with the text 'Filter cards'. The main content area is divided into two columns. The left column, titled 'Issue', contains three cards: 'Static Analysis - Code smell Type' (0 of 74 issues, opened by bicconi), 'Static Analysis - Vulnerability Type' (0 of 1 issues, opened by bicconi), and 'Static Analysis - Bug Type' (0 of 3 issues, opened by bicconi). The right column, titled 'In progress', shows a detailed view of a 'Static Analysis - Code smell Type' issue #149, opened in KUSV/SMA_T2 by user bicconi 11 seconds ago. It lists 'Code smell', 'Blocker - 1 Rules' (javaS1451 - 2 Components), 'Critical - 7 Rules' (including javaS121 - 97 Components, javaS1541 - 9 Components, javaS3776 - 6 Components, javaS2208 - 3 Components, javaS1192 - 4 Components, javaS3252 - 3 Components, and javaS1821 - 8 Components), and 'Major - 27 Rules'.

- System Testing (Testlink)



The screenshot displays the Testlink web application interface. The top navigation bar includes 'Testlink', 'My Jobs', and 'Test Project: 17 System Testing'. The main content area is split into two panels. The left panel, titled 'Test Case', shows a form for creating or editing a test case with fields for 'Test Case ID', 'Test Case Name', 'Priority', 'Responsible User', 'Step of Case', 'Last Date', and 'Status'. The right panel, titled 'Test Results on Test Case #10', shows a table of test results for a specific test case. The table has columns for 'Date', 'Status', 'Failed by', 'Notes', 'Exec (Days)', 'Module', and 'Pass'. Below the table, there is a section for 'Test Case #11 - Version: 1 - Test Case Files' and a 'Test Case Files' section with a table of file details.

Overall CTIP

