Software Quality Measurement

When creating software, the code should have the following characteristics:

1. The code should follow a specific convention
2. The code should be following established good practices and have been followed
3. Checked for potential bugs and performance, security, or vulnerabilities issues
4. Is the code duplicated anywhere
5. Does the code make logical sense, or is it too complex
6. Does the public API have good documentation and comments
7. Does the code have unit tests
8. Doe the code follow good software design and architecture principles.

We can enforce these coding standards automatically by two methods: Static code analysis or Dynamic code analysis. To explain them quickly:

Dynamic code analysis

Dynamic Code Analysis relies on studying how the code behaves during execution. The objective is to find errors in a program while it is running, rather than by repeatedly examining the code offline. Some things that Dynamic code analysis does are:

1. Code Coverage: Computing how much a piece of code gets tested by test suites
2. Memory error detection: Checking whether or not memory leaks or errors occur
3. Fault localization: Locating the buggy code to a specific location
4. Invariant Inference: Observes the values that the program computes, and then report properties that were true over the observed executions, and this likely true over all executions.
5. Security Analysis: Detect security problems.
6. Concurrency errors: Dynamic Uses runtime error detection to expose defects such as race conditions, exceptions, resource and memory leaks, and security attack vulnerabilities
7. Program slicing: Consists of reducing the program to the minimum form that still produces the selected behavior.
8. Performance Analysis: dynamically tracing software applications at runtime and captures data that can be used to analyze and identify the causes of poor performance.

Static Code Analysis

Static code analysis is done without executing any of the code. It is a collection of algorithms and techniques to analyze source code to automatically find potential errors and poor coding practices. This is done with compiler errors and run-time debugging techniques such as white box testing. Static code analysis is also considered a way to automate code review process. The tasks involved in static code analysis can be divided as such:

1. Detecting errors in programs
2. Recommendations on code formatting with a formatter
3. Metrics computation, which gives you back a rating on how well your code is.

Popular tools for static Code Analysis are **Checkstyle, PMD,** and **FindBugs**.

SonarQube Benefits

So Why SonarQube
So why not just existing and proven tools and configure them in the CI server ourselves? Well for SonarQube there are a lot of benefits:

- CI tools do not have a plugin which would make all of these tools work easily together
- CI tools do not have plugins to provide nice drill-down features that SonarQube has
- CI Plugins does not talk about overall compliance value
- CI plugins do not provide managerial perspective
- There is no CI plugin for Design or Architectural issues
- CI plugins do not provide a dashboard for overall project quality

Features of SonarQube are:

- Doesn't just show you what's wrong, but also offers quality and management tools to actively helps you correct issues
- Focuses on more than just bugs and complexity and offers more features to help the programmers write code, such as coding rules, test coverage, de-duplications, API documentation, and code complexity all within a dashboard
- Gives a moment-in-time snapshot of your code quality today, as well as trends of past and potentially future quality indicators. Also provides metrics to help you make the right decisions

Getting Started

Installation of Sonarqube:

Installing SonarQube in ubuntu

- Perform a system update and install unzip

```
sudo apt update
sudo apt install unzip -y
```

- Install Openjdk11

```
sudo apt install openjdk-11-jdk -y
```

- Install and Configure Postgres

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" >>
/etc/apt/sources.list.d/pgdg.list'
wget -q https://www.postgresql.org/media/keys/ACCC4CF8.asc -O - | sudo apt-key add -
sudo apt-get -y install postgresql postgresql-contrib
```

- Enable and Start Postgresql

```
sudo systemctl enable postgresql
sudo systemctl start postgresql
```

- Change the passwd for postgres user

```
sudo passwd postgres
```

- Switch to postgres user and create a user called sonar

```
su - postgres
createuser sonar
psql
```

- Set a password for the newly created user for SonarQube database and create a database for Postgresql database
-

```
ALTER USER sonar WITH ENCRYPTED password 'P@ssword';
CREATE DATABASE sonar OWNER sonar;
```

- Exit the psql shell and switch back to the user by running exit comand

```
\q
exit
```

- Download Sonarqube

```
wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-8.9.1.44547.zip
```

- Unzip the sonarqube using following command

```
sudo unzip sonarqube-8.9.1.44547.zip -d /opt
```

- Rename the directory

```
sudo mv /opt/sonarqube-8.9.1.44547 /opt/sonarqube
```

- Create a non sudo linux user

```
sudo adduser sonarq
```

- Assign permissions to sonarqube directory

```
sudo chown -R sonarq:sonarq /opt/sonarqube/
```

- Sonarqube uses the elastic search service so increase vm max map

```
sudo sysctl -w vm.max_map_count=262144
```

- Open the Sonarqube properties file `sudo nano /opt/sonarqube/conf/sonar.properties` and change the following properties

```
sonar.jdbc.username=sonar
sonar.jdbc.password=P@ssword
sonar.jdbc.url=jdbc:postgresql://localhost/sonar
sonar.web.javaAdditionalOpts=-server
```

- Configure Sonarqube as service

```
sudo nano /etc/systemd/system/sonar.service
```

- Add the following content to sonar.service

```
[Unit]
Description=SonarQube service
After=syslog.target network.target

[Service]
Type=forking

ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start
ExecStop=/opt/sonarqube/bin/linux-x86-64/sonar.sh stop

User=sonarq
Group=sonarq
Restart=always

[Install]
WantedBy=multi-user.target
```

- Now enable and start sonarqube

```
sudo systemctl enable sonar
sudo systemctl start sonar
sudo systemctl status sonar
```

- Now access the sonarqube with the ip address of the server `http://<ipaddress&gt;:9000`. Login into sonarqube with default credentails `username: admin` and `password: admin`