

Continuous Integration Tools

24/01/2017

Cécile Camillieri/Clément Duffau



Continuous Integration (previously)

Continuous:

“ Forming an unbroken whole; without interruption. ”

Integration:

“ The process of combining two or more things into one, to assemble in order to create an organic whole. ”



Continuous Integration:

The process to assemble without interruption

Continuous Integration (previously)

- Automate building of your product
- Have an overview of your system
- Trace the results of these activities

Today in Jenkins



Jenkins



maven

JUnit

Purpose : Automate compile and unit test at each commit

What can we
do more ?

Today in Jenkins



Jenkins



Purpose : Automate integration tests

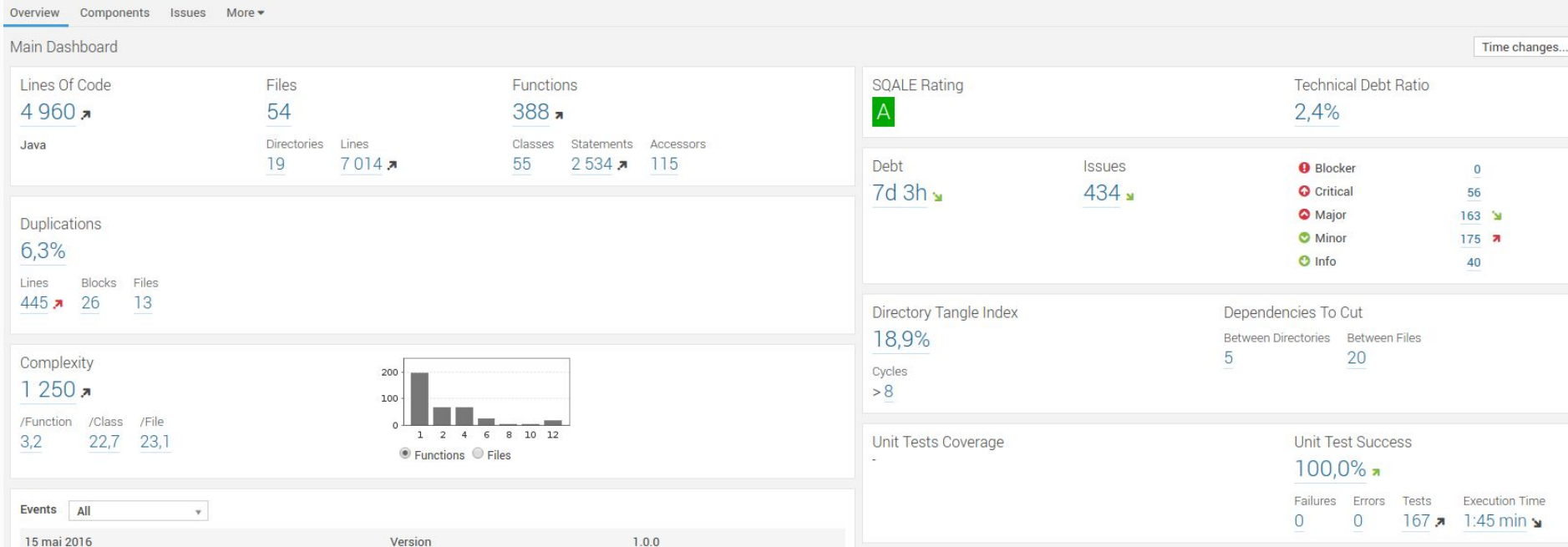


Purpose : Automate quality code analyze

SonarQube

- Quality metrics on your code
 - Cyclomatic complexity
 - Bug and vulnerabilities
 - Code smells
 - Duplication
- Test coverage
- Measurement of the technical debt and his evolution

SonarQube



SoapUI

- Services integration test framework
- Test services through HTTP to mock front-end calls
- Define test suite to check a scenario

SoapUI

The screenshot shows the SoapUI 5.2.1 interface. The top menu bar includes File, Project, Suite, Case, Step, Tools, Desktop, and Help. Below the menu is a toolbar with icons for Empty, SOAP, REST, Import, Save All, Forum, Trial, Preferences, and Proxy. A search bar for the forum and an online help link are also present.

The main workspace is divided into three panes:

- Navigator:** Shows a tree view of the test suite structure. A red box highlights the 'TestSuite' node, and a red arrow points from it to the right pane.
- Inspector:** Displays the details of the selected 'TestSuite'. A red box highlights the play button icon. Below it, a list of test cases is shown, all with green bars and the word 'FINISHED'.
- Properties:** A 'Custom Properties' section for 'TestSuite Properties' is visible at the bottom left, containing a table with 'Name' and 'Value' columns.

Property	Value
Name	TestSuite

At the bottom of the Inspector pane, there are tabs for 'Description', 'Properties', 'Setup Script', and 'TearDown Script'. Below these are sections for 'TestSuite Log' and various log files: 'SoapUI log', 'http log', 'jetty log', 'error log', 'wsrm log', 'memory log', and 'script log'.

Ok, we have new tools
but what about
continuous integration
?

Call to tools can be
automated !

SonarQube, Maven and Jenkins

```
<build>
  <pluginManagement>
    <plugins>
      <plugin>
        <groupId>org.sonarsource.scanner.maven</groupId>
        <artifactId>sonar-maven-plugin</artifactId>
        <version>3.2</version>
      </plugin>
    </plugins>
  </pluginManagement>
</build>
```

```
stage('back-maven') {
  sh 'mvn -f maven/pom.xml clean package'
  withSonarQubeEnv('Sonar') {
    sh 'mvn -f maven/pom.xml org.sonarsource.scanner.maven:sonar-maven-plugin:3.2:sonar'
  }
}
```

SonarQube in your project

- URL : <http://iut-outils-gl.i3s.unice.fr/sonar/>
- Documentation : <https://docs.sonarqube.org/display/SCAN/Analyzing+with+SonarQube+Scanner+for+Jenkins>
- SonarQube analyze for back-end AND mobile front-end
- Targeted metrics
 - A SQUALE
 - No blocker issues
 - No (or discussed with us) critical issues

SoapUI and Maven

```
<pluginRepositories>
  <pluginRepository>
    <id>smartbear-sweden-plugin-repository</id>

    <url>http://www.soapui.org/repository/maven2/</url>
  </pluginRepository>
</pluginRepositories>
```

```
<build>
  <plugins>
    <plugin>
      <groupId>com.smartbear.soapui</groupId>
      <artifactId>soapui-maven-plugin</artifactId>
      <version>4.6.1</version>
      <configuration>
        <!--The location of your soapUI setting file-->
        <projectFile>/home/test/test.xml</projectFile>
        <!--Where to place the output of the run-->
        <outputFolder>/home/test/output/</outputFolder>
        <!--Make the junit results file-->
        <junitReport>>true</junitReport>
      </configuration>
      <executions>
        <execution>
          <id>soapUI</id>
          <!--Run as part of the test phase in the Maven
lifecycle-->
          <phase>test</phase>
          <goals>
            <!--Run the test phase of maven-soapui-plugin-->
            <goal>test</goal>
          </goals>
        </execution>
      </executions>
    </plugin>
  </plugins>
</build>
```

```
mvn com.smartbear.soapui:soapui-maven-plugin:4.6.1:test ← just for SoapUI tests
```

```
mvn test ← for unit test and SoapUI tests
```

