**Software Testing and Quality Assurance Project Proposal**

**Title: Weka – Java Code Testing**

**Team Members:**

**Kshitija Joshi # M08766983**

**Mohammed Aamer FNU #M0880210**

**Instructor: Prof. Vignesh Subbian**

**Abstract:**

The goal of this project is to test SUT, Weka which is developed to use for data mining techniques and analyse the results generated from existing test cases. Weka is an open source software issues under GNU and programmed using Java

**Weka:**

It provides packages which presents a collection of algorithms for solving read world data mining problems. The software is written in Java and facilitates the graphical user interface to implement various machine learning techniques. Weka facilitates pre-processing of the data files such as arff, csv. Functions such as data normalization, conversion of data types of the data, adding missing values in the data types of transformation of data can be implemented in data. On the processed data, data mining techniques such as classification, clustering, association, visualization can be implemented on the data files. Weka also generates visual graphs and reports which helps to analyze pattern of the data.

The system has approximately 12007 lines of the code in Java, and contains modules for various techniques in data mining such as Classification, Association, Clustering and Data pre-processing. Also there are 18658 unit test cases available in the Weka source code. We are planning to analyse execution of test cases of modules such as Classifiers, associations, attribute selection which includes 2267, 228, 1068 test cases respectively.

Below is the proposed plan for this project to validate testing and quality of this system:

1. Software testing:

* **Coverage testing**- We are going to use *Emma* to find the code coverage such as statement coverage, branch coverage and analyze and interpret the results.
* **Unit Testing**- We are going to validate and verify the existing unit test cases. Also planning to use *FindBugs* , which detects a variety of common coding mistakes, including thread synchronization problems, misuse of API methods
* Perform manual software inspection on selected modules and further analyze to determine any correlation with results from static analysis.

1. Quality Assurance:

* Planning to use *JArchitect* to audit the code base before refactoring. It helps you also in your migration process and to analyze following aspects of quality assurance such as code quality and Statistical analysis.

*STAF***:** Planning to implement test automation using STAF which automates distribution, execution and result analysis of the test cases.

References:

[1] **Weka:** <http://www.cs.waikato.ac.nz/ml/index.html>

<https://weka.wikispaces.com>

[2] **FindBugs:** [http://findbugs.sourceforge.net/manual/eclipse.html](http://www.google.com/url?q=http%3A%2F%2Ffindbugs.sourceforge.net%2Fmanual%2Feclipse.html&sa=D&sntz=1&usg=AFQjCNHlcj-dvtED8pn10rITX1x5A4xYjw)

[3] **JArchitect:** <http://www.jarchitect.com/>

[4] **STAF Documentation:** http://staf.sourceforge.net/educ/STAF-STAX-HandsOn-Part1.ppt