|  |  |
| --- | --- |
|  | *Code Inspection Report*  *Anti-Spam Configuration Software Development Project*  BSc/MSc in [LEI | LIGE | METI]  Academic Year 2017/2018 - 1º Semester  Software Engineering I  Group 49  64671, Hugo Miguel, METIA1  65029, Inês Jorge, METIA1  66077, Maria Pires, METIA1  69300, Raquel Duque, METIA1  ISCTE-IUL, Instituto Universitário de Lisboa  1649-026 Lisbon  Portugal  December 12th 2017 |

**Table of Contents**

[Introduction 3](#_Toc498465002)

[Code inspection – Name of the component being inspected 3](#_Toc498465003)

[Code inspection checklist 3](#_Toc498465004)

[Found defects 3](#_Toc498465005)

[Corrective measures 3](#_Toc498465006)

[Conclusions of the inspection process 3](#_Toc498465007)

# ****Introduction****

The software developed aims to create new rules to face the threats, as well as the calibration of each rule for the final classification of the messages. The main objective of filter calibration is to provide an optimal configuration of the anti-spam filtering service that minimizes the occurrence of message classifications as legitimate (FP- False Positive), as on the other hand, it minimizes the occurrence of classifications of messages as legitimate when they are spam messages (FN- False Negatives).

# Code inspection – Name ****of the component being inspected****

The software component that we are going to inspect will be 4 classes implemented, in Java, for the development of the intended software, Anti-Spam Configuration Software Development Project.

|  |  |
| --- | --- |
| *Meeting date:*  *Meeting duration:*  *Moderator:*  *Producer:*  *Inspector:*  *Recorder:* | *12/12/2017*  *30 minutes*  *Inês Jorge*  *Hugo Miguel, Inês Jorge, Maria Pires, Raquel Duque*  *Maria Pires, Raquel Duque*  *Hugo Miguel* |
| *Component name (Package/Class/Method):* | *GUI, FileManager, Rule, Engine* |
| *Component was compiled:* | *Yes* |
| *Component was executed:* | *Yes* |
| *Component was tested without errors:* | *Yes* |
| *Testing coverage achieved:* | *86.3%* |

# Code inspection checklist

The checklist for Java code inspection used in this project is available at <http://www.cs.toronto.edu/~sme/CSC444F/handouts/java_checklist.pdf> and in blackboard ES1 page.

# Found defects

Identify and describe found defects, opinions and suggestions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Found defect Id** | **Package, Class, Method, Line** | **Defect category** | **Description** |
| 1 | antiSpamFilter, GUI, updateGUI(),136 | 11 | More than 60 lines |
|  |  |  |  |
| 2 | antiSpamFilter, GUI, createTable(), 211  antiSpamFilter, FileManager, FileManager(), 32  antiSpamFilter, FileManager, readRf(), 161  antiSpamFilter, FileManager, readRs(), 187 | 13 | Object and array references not null once the object or array is no longer needed |

# Corrective measures

1. We split the methods code into new methods and used recursion.

2. A function that has set all arrays to null has been created. To do this, we use booleans to only happen when the arryas are no longer needed.

# Conclusions of the inspection process

We have variables with similar names because we divide the interface into 2, each part for a configuration type, and both parts use the same attributes and methods.