Developing an Automated Testing Framework for OpenMRS

Ricky Ramos, Matthew Schwarz, Matthew Kay
Computer Science Department
College of Charleston

Abstract

Testing code is an imperative part of the software engineering process. This is done in order to ensure that robust, high quality software is being built that meets the customer's requirements. This work shows the benefits of having an automated testing framework to test various methods in OpenMRS, an open-source electronic medical record system platform.

Introduction

OpenMRS is an electronic medical record system platform written in the Java programming language that is supported by a community passionate about healthcare. They are mainly concerned with aiding the developing world where AIDS, tuberculosis, and malaria afflict the lives of millions of people. Our task this semester was to develop an automated testing framework to test five of their methods.

```
/TestAutomation
    /project
        /src
        /bin
        / …
    /scripts
        runAllTests.(some scripting extension)
        … other helper scripts
    /testCases
        testCase1.txt
        testCase2.txt
        …
    /testCasesExecutables
        testCase1(may be folder or file)
        …
    /temp (for output from running tests … be sure to clean at start of runAllTests)
        testCase1results (might be folder or file)
        …
    /oracles
        testCase1Oracle (might be folder or file)
        …
    /docs
        README.txt
    /reports
        testReport.(txt or html)
```

Figure 1: Framework Directory Structure

```
# Test cases are written in the following format:
# 1. Test Case ID
# 2. Requirement being tested
# 3. Component being tested
# 4. Class
# 5. Driver being used
# 6. Test Inputs(s)
# 7. Expected Result
# 8. Package
```

Figure 2: Example Test Case Format

Methods:

1. Create the framework directory structure from Figure 1 in the team Github repository.
2. Clone the OpenMRS repository and upload it to the /TestAutomation/project/src folder within the team Github repository.
3. Clone the team Github repository to our local Ubuntu Linux machines.
4. Select 5 methods from the OpenMRS source code to test. The five methods chosen were:
   compareNaturalAscii(String s, String t)
   isStringInArray(String str, String[] arr)
   containsOnlyDigits(String str)
   containsDigit(String str)
   compareNaturalIgnoreCaseAscii(String s, String t)
5. Construct 5 test cases for each method using the format in Figure 2 and save them within the testCases folder.
6. Develop .java drivers for each method.
7. Develop a bash script that parses the lines of the test case files, compiles and runs the drivers, and outputs the test results into the terminal and a browser.
8. Perform 5 fault injections (1 fault per method).

```java
package org.openmrs.util;

public class OpenmrsUtilDriver2 {
	public static void main(String[] args) {
		String[] temp = args[0].split(", ");
		if(temp.length == 1) {
			String str = temp[0];
			System.out.println(OpenmrsUtil.containsOnlyDigits(str));
		}else {
			System.out.println("Error: Incorrect number of arguments!");
		}
	}
}
```

Figure 3: Example Driver

```java
public static boolean isStringInArray(String str, String[] arr) {
  //Correct: boolean retVal = false;

  //***FAULT***//
  boolean retVal = true;

  if (str != null && arr != null) {
    for (int i = 0; i < arr.length; i++) {
      if (str.equals(arr[i])) {
        retVal = true;
      }
    }
  }
  return retVal;
}
```

Figure 4: Example Fault

Results:

Team 404-Error Openmrs Testing Results

Test	Requirement	Method	Class	Driver	Inputs	Expected Output	Result
Test #001	This is testing that the compareNatural method will return a 0 if the two strings are equal.	compareNaturalAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver	test, test	0	0
Test #002	Testing that the compareNatural method will return a negative number if s lexicographically precedes t	compareNaturalAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver	test, white	-3	-3
Test #003	Testing that the compareNatural method will return a positive number if s lexicographically follows t	compareNaturalAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver	test, string	1	1
Test #004	Testing that the compareNatural method will return a positive number if s lexicographically follows t	compareNaturalAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver	test, check	17	17
Test #005	Testing that the compareNatural method will return a positive number if s lexicographically follows t	compareNaturalAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver	test, pneumonia	4	4
Test #006	This is testing that the correct boolean value will be printed from the isStringInArray(String str, String[] arr) method in OpenmrsUtil.java	isStringInArray(String str, String[] arr)	OpenmrsUtil.java	OpenmrsUtilDriver	random, random, test, computer	true	true
Test #007	This is testing that the correct boolean value will be printed from the isStringInArray(String str, String[] arr) method in OpenmrsUtil.java	isStringInArray(String str, String[] arr)	OpenmrsUtil.java	OpenmrsUtilDriver	test, random, test, computer	true	true
Test #008	This is testing that the correct boolean value will be printed from the isStringInArray(String str, String[] arr) method in OpenmrsUtil.java	isStringInArray(String str, String[] arr)	OpenmrsUtil.java	OpenmrsUtilDriver	computer, random, test, computer	true	true
Test #009	This is testing that the correct boolean value will be printed from the isStringInArray(String str, String[] arr) method in OpenmrsUtil.java	isStringInArray(String str, String[] arr)	OpenmrsUtil.java	OpenmrsUtilDriver	allergy, random, test, computer	false	false
Test #010	This is testing that the correct boolean value will be printed from the isStringInArray(String str, String[] arr) method in OpenmrsUtil.java	isStringInArray(String str, String[] arr)	OpenmrsUtil.java	OpenmrsUtilDriver	medicine, random, test, computer	false	false
Test #011	This is testing that the correct boolean value will be printed from the containsOnlyDigits(String str) method in OpenmrsUtil.java	containsOnlyDigits(String str)	OpenmrsUtil.java	OpenmrsUtilDriver2	12345	true	true
Test #012	This is testing that the correct boolean value will be printed from the containsOnlyDigits(String str) method in OpenmrsUtil.java	containsOnlyDigits(String str)	OpenmrsUtil.java	OpenmrsUtilDriver2	97930	true	true
Test #013	This is testing that the correct boolean value will be printed from the containsOnlyDigits(String str) method in OpenmrsUtil.java	containsOnlyDigits(String str)	OpenmrsUtil.java	OpenmrsUtilDriver2	99999990	true	true
Test #014	This is testing that the correct boolean value will be printed from the containsOnlyDigits(String str) method in OpenmrsUtil.java	containsOnlyDigits(String str)	OpenmrsUtil.java	OpenmrsUtilDriver2	This is a string!	false	false
Test #015	This is testing that the correct boolean value will be printed from the containsOnlyDigits(String str) method in OpenmrsUtil.java	containsOnlyDigits(String str)	OpenmrsUtil.java	OpenmrsUtilDriver2	1234123412346	false	false
Test #016	This is testing that the correct boolean value will be printed from the containsDigit(String str) method in OpenmrsUtil.java	containsDigit(String str)	OpenmrsUtil.java	OpenmrsUtilDriver3	test87	true	true
Test #017	This is testing that the correct boolean value will be printed from the containsDigit(String str) method in OpenmrsUtil.java	containsDigit(String str)	OpenmrsUtil.java	OpenmrsUtilDriver3	666666659	true	true
Test #018	This is testing that the correct boolean value will be printed from the containsDigit(String str) method in OpenmrsUtil.java	containsDigit(String str)	OpenmrsUtil.java	OpenmrsUtilDriver3	11234	true	true
Test #019	This is testing that the correct boolean value will be printed from the containsDigit(String str) method in OpenmrsUtil.java	containsDigit(String str)	OpenmrsUtil.java	OpenmrsUtilDriver3	string	false	false
Test #020	This is testing that the correct boolean value will be printed from the containsDigit(String str) method in OpenmrsUtil.java	containsDigit(String str)	OpenmrsUtil.java	OpenmrsUtilDriver3	asdfasdkajsdf	false	false
Test #021	This is testing that the compareNatural method will return a 0 if the two strings are equal.	compareNaturalIgnoreCaseAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver2	test, test	0	0
Test #022	Testing that the compareNatural method will return a negative number if s lexicographically precedes t	compareNaturalIgnoreCaseAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver2	Test, wiiTe	-3	-3
Test #023	Testing that the compareNatural method will return a positive number if s lexicographically follows t	compareNaturalIgnoreCaseAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver2	test, aziNG	1	1
Test #024	Testing that the compareNatural method will return a positive number if s lexicographically follows t	compareNaturalIgnoreCaseAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver2	TEST, Check	17	17
Test #025	Testing that the compareNatural method will return a positive number if s lexicographically follows t	compareNaturalIgnoreCaseAscii(String s, String t)	NaturalStrings.java	NaturalStringDriver2	Hlfe, pneUMOnia	4	4

Figure 5: Test Case Results Before Fault Injections

Conclusion:

We were able to successfully build an automated testing framework for OpenMRS that displays the testing results in a HTML table that appears in a browser as well as a terminal. In Figure 5 above, the table displays the test number, requirement, method, class, driver, inputs, expected output, and the result for each test case. All test cases were initially designed to pass, but the introduction of fault injections caused some of the test cases to fail, which was expected.

Acknowledgments:

We would like to thank Dr. Jim Bowring for his constructive criticism and guidance throughout this entire project.