Codmon: A multi-platform modular test environment.

Berend van Veenendaal

February 19, 2014

TODO:Abstract

Preface

TODO: Preface,acknowledgements

Contents

1 Introduction

This chapter will give an introduction about the Codmon 2.0 project by giving a brief decription of background of my research and of the previous version of the Codmon project. It also describes the structure of the reminder of this thesis.

1.1 Background

In times when software projects become more and more complex, testing of this software becomes more and more important. Many software related problems are caused by lack of testing of the software [2]. One of the challenges of software engineering is to make sure that the software behaves in the same way on different platforms. Even when software is written in such a way that it can run on multiple platforms, there are still issues that must be dealt with, before one is able to run and test the software. Think about the configuration of the test environment or finding and installing all the prerequisite libraries etc etc.

1.2 Problem indication

TODO paper references for Hudson, Jenkins, Junit and CsUnit TODO Expand problem indication tell more about HUdson,Jenkings Junit

Now days there are numerous test frameworks and test environments available. For example there is *Junit* for Java-unit testing and *csUnit* for C#-unit testing. There are also different environments like Hudson, Jenkins which can build a project and run a series of (unit) tests against this project. All of the frameworks and environments have bother there advantages and disadvantages. One of the advantages of unit testing is that a software developer easily can add new *functinal* unit tests. One of the disadvantages is that standard unit testing ignores non-funtional tests like performance testing. Jenkins and Hudson,like Unit tests, also have their disadvantages. One of them is, althoug they both run on different platforms, in their usage they are not realy platform independent. For example, If you want to make a connection form Hudson or Jenkins to a remote machine you need to know in advance on which platform this remote machine runs.

1.3 Problem statement

The test frameworks and test environments described in section 1.2 can be criticized on one or more aspects. What we are looking for is infact, a combination of the positive aspects of the described frameworks and environments, without getting the undesirable aspects for free. So the big question is, is it possible to design a multi-platform, modular test environment? If the answer to this question is yes, if we manage to design such a test environment, is it also possible to design is in such a way that is user friendly and maintainable? To be able to answer the second question we first have to define what is meant with *"user friendly"* in this thesis. With user friendly we mean that is must be possible to add easily both new test cases and software under

test to the test environment. In other words a user must be able to add both new test cases and software to the test environment without nowing anything about the internal mechanisms of the test environment.

This thesis describes a multi-platform modular test environment called Codmon 2.0. The Codmon 2.0 project provides a set of virtual machines, in which Codmon 2.0 is aleady installed and preconfigured.

First, to run on multiple platforms the software must be written in a platform independent language, for example Java [1].

TODO: Describe the problem (refer to other environments/platforms) TODO: describe research questions.

1.4 Thesis outline

TODO: Describe whats where in this thesis

2 Codmon

2.1 the road to codmon

//TODO: Think of better subsection title!! //TODO: Explain ideas and road to solution

2.2 Codmon 2.0

//TODO: adapt Codmon 2.0 to new project name. //TODO: 2) Explain Solution, including why it's different then existing solutions (compare with solutions described in the problem indication)

3 Conclusion

3.1 conclsuion

TODO: give answers to the questions from section Problemstatement TODO: Discus related future work

References

[1] "Henry McGilton" "James Gosling". The java language environment: Contents. May 1996. Section 4.2.

[2] "Masato Shinagawa" "Toshiaki Kurkowa". Technical trends and challenges of software testing. *Science and technology trends*, 29, 2008.