# TJHSST Computer Systems Lab Senior Research Project
# Automated System Testing
# 2008-2009

Ian Garrett

April 2, 2009

**Abstract**

Around the world, many companies require testing for various projects. This process can take numourous hours of manual labor, which in turn costs companies a lot of money. With automated software, companies do not need to spend human resources to manuelly test. In the business world, there is not much room for time or money to be wasted. This project reduces the time spent in testing with the use of software that can be obtained easily. It sets up a system in which one client system can test multiple applications on many server systems. The one client system implements various automation tools to accomplish the task. The practical use of this project is to reduce the time spent on testing while using automation tools that anyone can obtain. The project will show that the manual testing that originally takes hours to test on the one or two other systems can be reduced to just minutes using one system.

**Keywords:** automation, client, server, testing

## 1 Introduction

### 1.1 Scope of Study

In the process of testing, this project implemented a few significant applications. TightVNC and VNCRobot play critical roles within the project. I have set up the TightVNC, which will create either a server or a client onto the system. The server will make available the server's desktop for the client machine. When the client opens up a viewer, the client will be able to access all of the servers that it is connected to. This lets the VNCRobot, which is an automation tool used to automate simple actions, control the server systems.

Although this project can be done with many systems, this project will be limited to two systems. This is to not overwhelm the project as setting up systems may take a long time, and the main objective of this to prove automation is effective as a tool by reducing test time, not expansion into many systems.

TightVNC, creates a VNC envirnonment. This is an application that is not specfically designed for automation, but just for any purpose of use through VNC. Having a VNC environment means that two (or more) computers are connected and be accessed remotely. The importance of this application is to be able to access another system from a single system. TightVNC (as well as any VNC environment) invovles two components. The first is the client, and the second is the server. Server systems are controlled by the client system, which is responsible for any work being done remotely. While alone TightVNC is not automation, it plays a vital role in the automation process in this experiment.

The other application that is key for success in this project is VNCRobot. Like TightVNC, VNCRobot implements a VNC environment. The difference is in that VNCRobot can be used to control another system. By itself, VNCRobot is not test automation as it can be used (as is more commonly used) for other purposes. VNCRobot is significant in my project because it has a feature where commonds for movements in the other system can be programmed. Using VNCRobot as well as coding in the movements for testing helps create the automation.

A advantage to the method I use to create the test automation is the ease in which it can be created. The applications that help with the automation are easy to obtain. This creates a cost effective means to automate tests for future software.

## 1.2 Expected results

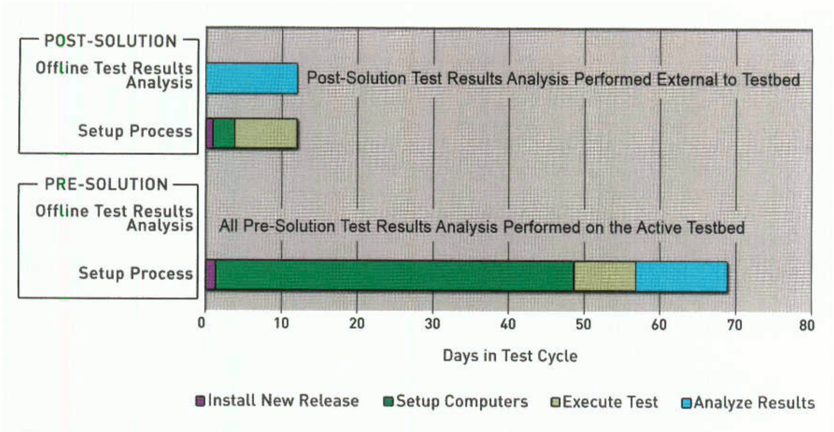
This project is expected to have a reduced testing time using automation. The system that is set up should include a client with a server and implement easily accessible applications during the process. There should be a drastic difference between the manual testing time and the automated testing time.

This project may be later extended to a large-scale system. If the client were to control a hundred servers, instead of another one, it may change the outcome of the project. Therefore, the testing of systems through automation can prove to have even more purposes.

# 2 Background and review of current literature and research

Companies have been trying to impletment automated testing within their companies for years. Many common problems that halt the continuation of this is a lack of knowledge in the area and the long time that it takes to set up automation. Years ago, automated testing was not practical for use due to the long times it took to set up an automation system. Currently, there are many tools for automation that makes automated system testing not only simple but extremely quick to set up. The increased knowledge and simplicity of the automation process in key in automated system testing.

Figure 1: Gonzalez ”Automating Software Configuartions in Test Labs”



A similar experiment to this project was done by Gonzalez and Spade. They also chose to attempt in reducing testing time to reduce cost of testing. However, they did not use easily accessible means of testing. The data (figure 1) shows that the means in which they tested did in fact reduce the time from almost 70 hours of total set up and testing to slightly over 10 hours to do so. My project will not include set up time as this one did.

Often people do not realize the significance of automation and overlook it. Unfortunetely, when people get around to testing there are many obstacles that may occur. The paper titled ”Heuristics-based infeasible path detection for dynamic test data generation” realizes the importance of automated tests and its role in reducing cost and increasing reliability of software testing. It also states that there is a large challenge with path-oriented test data generation. The goal of the experiment was to find a way to solve the issue of infeasible program path detection for dynamic test data generation. A key player in solving the issue is the fact that many of the infeasible program paths have similar properties. They use this fact to try and solve the problem by knowing what signs would most likely lead to an infeasible path. This way, the path can be detected with fairly high accuracy.

Another paper titled "Building test cases and oracles to automate the testing of web database applications" explains how many organizations use web applications with databases to store data but manuel testing of these applications can take an impractical amount of time. The solution to this is to automate the testing but this can even prove to fail due to poor test cases, even if the automation code is perfect. Therefore, the Automatic database Tester was created to generate functional test cases that would be used for database updates. The project that I am doing uses a test case to set direction in the project.

# 3 Procedures and Methodology

Various applications were implemented in the automation of the systems. In addition, there were applications that were tested to prove that the automation worked correctly. For example, SeaClear, which is a navigation tool, was used. There are many drop down menus and well as places where values can be input. For example, known valid values were plugged in. Later, known invalid values were plugged in to see if an error menu will appear. After testing this applications manuelly, the client system will then attempt to recreate the process at a faster speed. Also, the project will be tested by observing if the different systems interact correctly with each other. If the applications can not even be tested, it means the framework was not correctly set up.

The client system will use TightVNC to connect to the server systems. This will create a relationship between one client and one server. I have coded commands into VNCRobot so that not only is the server system accessed, but it is now automated. The SeaClear application (figure 2) will be tested and the results will be sent back to the client. The test commands have been added in so that the process is now an automated test of SeaClear. The operator, will then be able to view the results within minutes of starting the testing. The data collection will be through the comparison between manual and automated testing. The initial test will be a test case built for a manual test whose time will be recorded. This includes work in SeaClear.

The project's accuracy will be tested by first manually testing the application. After the results have been recorded, the client will attempt to recreate teh results using the same testing process. If the results match, the project was successful. This process will be repeated with different types of inputs, such as valid verses invalid inputs, to show that any input will produce correct results.

The data collection will be through the comparison between manual and automated testing. The initial test will be a test case built for a manual test whose time will be recorded. This includes work in SeaClear.

The next step in the process is to automate the test case and then run the test case with the automated environment that has been set up. The time for completion of this step will also be recorded.

# 4 Results

This project is expected to create a system in which applications on many servers can be tested through the use of one client. Although this can be done manually, the project is expected to reduce testing time from hours to minutes. In addition, the automation of the testing will not compromise the accuracy of the testing in any way.

This project can be expanded by the use of more servers. If the project were to be put on a grander scale, nothing should change but there is a possiblity.

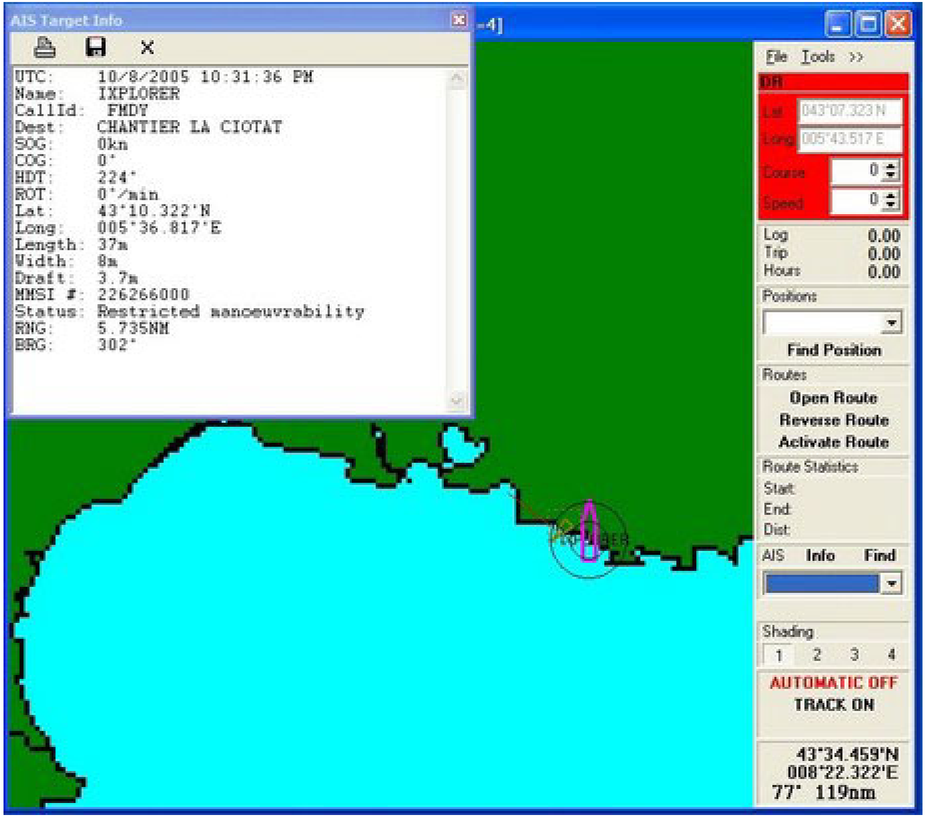
# 5 Analysis

The data collected was in the form of different tests. The main objective was first to have everything running properly. After setting up the environment that was needed to have everything working, I started to test different application before settling for the one that would mainly be used for the project.

One of the simple tests was to see how the automation would work with a word processor. To test this, I had the program open up Microsoft Word and type out the log that was due that day. Through this test I learned that the program could work for a long period of time. This was shown through the fact that I had a full-page log and the program successfully complete it.

Another simple test that was used was to take a screenshot of the stocks and email them to myself then check it. I went to BigCharts.com to check a random stock then took a screenshot. Then, I entered my email and sent the file of the picture to myself. I did this multiple times. I learned from this that the program could be used to repeat tasks, which was good because that was one of the projects main goals.

Image of the SeaClear Program



```
UTC:      10/8/2005 10:31:36 PM
Name:     IXPLORER
CallId:    FMDY
Dest:     CHANTIER LA CIOTAT
SOG:      0kn
COG:      0°
HDT:      224°
ROT:      0°/min
Lat:      43°10.322'N
Long:     005°36.817'E
Length:   37m
Width:    8m
Draft:    3.7m
MMSI #:   226266000
Status:   Restricted manoeuvrability
RNG:      5.735NM
BRG:      302°
```

DR
Lat 043°07.323 N
Long 005°43.517 E
Course 0
Speed 0

Log 0.00
Trip 0.00
Hours 0.00

AUTOMATIC OFF
TRACK ON

43°34.459'N
008°22.322'E
77° 119nm

Figure 2

# 6 Discussion and Conclusion

The results that were expected of this project were for the most part true. A system for multiple platforms was created using a VNC Server, VNC Client, and VNC Robot. The main issue was trying to implement the VNC Server and VNC Client with Linux. At first it seemed as if Linux would not be compatible with it but it was later determined that it could be. It was particularly interesting to see Windows and Linux being used to automate the other.

# 7 Referances

Dhavachelvan, P., Uma, G., Venkatachalapathy, V. (2003, November 14). A new approach in development of distributed framework for automated software testing using agents.

Gonzalez, R., Spade, R. (n.d.). Automating Software Configuartions in Test Labs.

Hawkins, J., Nguyen, H., Howard, R. B. (n.d.). Automated Real Time Testing (ARTT) for Embedded Control Systems (ECS).

Ngo, M. N., Tan, H. B. K. (2007, January 15). Heuristics-based infeasible path detection for dynamic test data generation.

Ramamoorthy, C., Ho, S. (1975). Testing Large Sotfware with Automated Software Evaluation Systems. Retrieved September 30, 2008

Ran, L., Dyreson, C., Andrews, A., Bryce, R., Mallery, C. (2006, May 24). Building test cases and oracles to automate the testing of web database applications. Retrieved October 28, 2008