**Validation Plan for *Automated Test Tool for GUIs***

**ASRC Federal Mission Solutions**

|  |  |
| --- | --- |
|  |  |
| Prepared by: | *Donald Zellman* |
| Date: | *11-06-2019* |
| Version: | 1.0 |

Document Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Description** |
| 1.0 | *Donald Zellman* | *11-06-2019* | Initial Release |

Table of Contents

* **Introduction**
* **Objective**

This document describes the Automated Test Tool for GUIs program. The idea behind the automated test tool that is being developed, is to drastically increase the speed and efficiency of the testing abilities of the customer.

* **Scope**

The primary goal of the Automated Test Tool for GUIs is to be able to assist in increasing the efficiency of testing for the ASRC display system department. By designing an interface in Motif that is able to record the user action and saving them into an XML file. After a C++ tool will read the XML and provide it to Google Test. Google Test will mock the object to perform the automated test. As well, to perform the test without opening the GUI. Most importantly, the Google Test will perform the test regardless of monitors' attributes.

* **References**

Definitive guide to the X Window guide: Motif Programming Manual vol. 6A Dan Heller and Paula M. Fergusa

Definitive guide to the X Window guide: X Toolkit Intrinsics programming manual V4 By Adrian Nye and Tim O'Reilly

UIM/X Beginner Guide by Integrated Computer Solutions, Inc. 2005-2007

Brochard, Adrien. “How to Simulate Key Press and Mouse Movement in Linux.” Xmodulo, 10 July 2014, <http://xmodulo.com/simulate-key-press-mouse-movement-linux.html>.

Sissel, Jordan. “Xdotool - Fake Keyboard/Mouse Input, Window Management, and More.” Semicomplete, 24 Apr. 1970, <https://www.semicomplete.com/projects/xdotool/>.

* **System Description**

The following is a high-level description of the system requirements and design choices made to fulfil them:

System Requirements:

Minimum:

Requires a 64-bit processor and operating system

OS: 64-bit Linux flavors involving CentOs

Processor: Intel pentium or Equivalent

Memory: 2 GB RAM

Graphics: Intel HD graphics

XWindow: Version 11

Storage: 1 GB available space

Recommended:

Requires a 64-bit processor and operating system

OS: 64-bit Linux flavors involving CentOs

Processor: Intel i3-3220 or Equivalent

Memory: 4 GB RAM

Graphics: AMD Radeon HD 7750, NVIDIA GeForce GTX 650 or Equivalent

XWindow: Version 11

Storage: 2 GB available space

User Capabilities

* Users will be able to test code using commands in the terminal using red hat linux.
* Users will be able to edit code using codeblocks in order to test what they need.
* Users will be able to save time and be more efficient with this automated test tool.

An in-depth description of all these high-level requirements can be found in the Requirements Document. To meet these system requirements, the following design decisions were made:

The *test tool* is compatible with *red hat linux* machines running with *red hat versions 7 or 8* with *google test installed.*

* **Validation Activity Requirements**
* **Validation Approach**

Our approach to validation is using unit testing to ensure each component of the *Automated Test Tool for GUIs* functions as intended. We plan to test *list what you plan to test.* Testing will be done by the development team.

* **Validation Methodology**

The following sub-sections describe the methodology utilized to validate the system.

* Installation Qualification (IQ)

The Installation Qualification will validate the components of the *Automated Test Tool* related to the installation and setup of the program software on compatible workstation computers. The system will be qualified when the following conditions have been carefully tested and met:

* *Google Test* is installed on the machine
* *Automated Test Tool* is installed on the machine
* Code Blocks is installed as the IDE.
* Operational Qualification (OQ)

The Operation Qualification will validate that a user with a compatible workstation is able to utilize all the functionality and interactivity of the *Automated Test Tool*. The standard use cases that must be carefully tested for the operation qualifications of the application to be approved are defined below:

* User can input arguments to run *the test tool* via command line such as “./main” or “./Test”.
* User can edit code using the IDE of codeblocks in C language.

The configured system is the system to be validated during OQ, which will be performed according to pre-approved test plans with defined acceptance criteria, using a qualified data set. OQ will generate documented test results that are subject to review and approval. All the OQ test documentation will form part of the validation package. OQ will confirm that the computerized processes perform their required function and meet the agreed business requirement and Functional Requirement Specification.

* Performance Qualification (PQ)

To ensure that user stories can be successfully completed as part of the application, our team has conducted the following performance qualification tests:

|  |  |
| --- | --- |
| **User Story** | **Validated By** |
| User’s ability to save time and money with this tool | An interface which accepts command line arguments |
| User’s ability to run our test with no issues. | Our various tests that we have run. |

* **Assumptions, Exclusions and Limitations**
* **Assumptions**

It is assumed that the *Automated Test Tool* is installed and ready to run on the user’s machine/workstation. Also, we assume that *igoogle test and red hat linux version 7 or 8* is installed on the machine.

* **Exclusions**

Nothing is excluded from validation.

* **Limitations**

There are no limitations.

* **Testing Environment**

The Development team will be using *the terminal from the red hat linux machine, and google test*  to unit test each component.

* **Traceability Matrix**

See the *Automated Test Tools* Traceability Matrix document.