# Introduction

Defense budgets are growing tighter and system/software developers are being pressured to create higher-quality products for less money. As a result, it is becoming increasingly important to find ways to reduce costs and improve the efficiency of testing.

Testing has historically been a manual effort requiring extensive human interaction with the system under test (SUT). The high cost of human labor can drive programs to reduce the overall amount of testing performed. Although this reduces costs, it also increases the risk that problems will not be discovered. The sooner defects are found in the life cycle of a program, the cheaper and easier they are to fix.

Test automation can provide a solution by reducing the manpower necessary to perform exhaustive product testing at all levels.

## Problem Space

There are two general approaches to test automation:

* Code-driven testing - The interfaces to classes, modules or libraries are tested with a variety of input arguments to validate that the results that are returned are correct.
* Graphical user interface (GUI) testing - A testing framework generates user interface events such as keystrokes and mouse clicks, and observes the changes that result in the user interface, to validate that the observable behavior of the program is correct.

The intent of this paper is to focus on the GUI testing framework tools currently available on the market. A variety of tools exist for this task, both proprietary and open source.

## Mission

Evaluate several of the most commonly used Automated GUI Testing tools in order to gain experience with them, compare and contrast their features, and assess their strengths and weaknesses as a whole.

## Executive Summary

*TBD – add summary of our findings and conclusions here*