# Software Requirements Specification
for the
# Track and Control System

Version 1.0

Robert Moss, Aaron Periera, Matthew Shrago

February 4, 2014

# Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

| **Signature** | **Printed Name** | **Title** | **Date** |
|---|---|---|---|
| | | | |
| | | | |
| | | | |

# Contents

# 1 Introduction

## 1.1 Purpose

The Software Requirement Specification (SRS) for the Track and Control System will explain in detail necessary features that the client purposes and the developers provide. The user base will be focused on anyone who will want to use their computer in a more efficient way and take advantage of their full screen space.

## 1.2 Scope

1. Track and Control System
   (a) The software will track a user's movements and be able to control numerous features around their Desktop with the movement of their head.
      i. The everyday computer user will utilize this software to organize their cluttered Desktop.
   (b) The application will also act as a security monitor to recognize when you're present at your computer and lock your screen accordingly.
      i. A benefit of this will be a sense of security for the user when they're away from their computer.

## 1.3 Definitions, Acronyms, and Abbreviations

- Application Specific Definitions
  - TACS - Track and Control System
  - TM - Tracking Module
  - WCM - Windows Control Module
  - SM - Settings Module
- Industry Definitions
  - OpenCV - Open Computer Vision: An open source library for object tracking via the camera.
  - SQLite - A lightweight, low maintenance, self contained database.
  - RGB - Red, Green, Blue color values.
  - HSV - Hue, Saturation, Value.
  - API - Application Programming Interface
  - C++ - An object oriented programming language.
  - GUI - Graphical User Interface
  - QT - An API for building GUIs

### 1.4 References

The list of references below are software documentation that we will be using:

1. OpenCV documentation: http://opencv.org/
2. Windows API Index: http://msdn.microsoft.com/en-us/library/hh920508(v=vs.85).aspx
3. QT C++ documentation: http://qt-project.org/

### 1.5 Overview

The rest of the SRS will contain:

1. Specific features of TACS and their details.
2. System Requirements
3. Design Constraints for the application.
4. Risks within the scope.
5. Any additional information about the development of the application.

## 2 General Description

This section will explicitly lay out the product's functionality and constraints as well as compare it to existing products. Possible set backs will be discussed. Details about each requirement will be in the Specific Requirements section of the SRS.

### 2.1 Product Perspective

A vast array of other products that utilize OpenCV are available, but no product will control your Desktop with the movement of your head. There has been much research in the field of physical interactions between the user and their computer. However, the application of organizing and securing your Desktop with this interaction is the novel portion of this software.

### 2.2 Product Functions

There will be many available functions the users can utilize. The modular design of the software will allow for additional functions to be easily implemented. *It should be noted that the window functions will be activated with a user-defined hot key.*

1. A proportional snap to grid set up for users to place desired windows in five different locations. Namely, left, right, top, bottom, and middle. Once all the windows are selected, the middle window will fill the screen and the user will be able to "peer" in the direction of the four other windows to show a preview of them.
2. The software will be able to detect when the user is away from the screen, and with a user defined delay-time, be able to lock or put your computer to sleep (all settings can be changed from the SM).

3. The software will organize all open windows into a 3D view (with the illusion of layered windows) for the user to "look" around their desktop and see which window they want to select.

### 2.3 User Characteristics

Any computer user who has trouble with screen space will find this software useful. The general user will be a laptop owner who has a built in camera.

### 2.4 General Constraints

This subsection of the SRS should provide a general description of any other items that will limit the developers options for designing the system. (See the IEEE Guide to SRS for a partial list of possible general constraints).

### 2.5 Assumptions and Dependencies

This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.

## 3 Specific Requirements

This will be the largest and most important section of the SRS. The customer requirements will be embodied within Section 2, but this section will give the D-requirements that are used to guide the projects software design, implementation, and testing.

Each requirement in this section should be:

- Correct
- Traceable (both forward and backward to prior/future artifacts)
- Unambiguous
- Verifiable (i.e., testable)
- Prioritized (with respect to importance and/or stability)
- Complete
- Consistent
- Uniquely identifiable (usually via numbering like 3.4.5.6)

Attention should be paid to the carefuly organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

### 3.1.2 Hardware Interfaces

### 3.1.3 Software Interfaces

### 3.1.4 Communications Interfaces

## 3.2 Functional Requirements

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

### 3.2.1 Functional Requirement or Feature 1

#### 3.2.1.1 Introduction

#### 3.2.1.2 Inputs

#### 3.2.1.3 Processing

#### 3.2.1.4 Outputs

#### 3.2.1.5 Error Handling

### 3.2.2 Functional Requirement or Feature 2

## 3.3 Use Cases

### 3.3.1 Use Case 1

### 3.3.2 Use Case 2

## 3.4 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95 of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, ¿ 30 day MTBF value, etc).

#### 3.4.1 Performance

#### 3.4.2 Reliability

#### 3.4.3 Availability

#### 3.4.4 Security

#### 3.4.5 Maintainability

#### 3.4.6 Portability

### 3.5 Inverse Requirements

State any *useful* inverse requirements.

### 3.6 Design Constraints

Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

### 3.7 Logical Database Requirements

Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc.

### 3.8 Other Requirements

Catchall section for any additional requirements.

## 4 Project Planning and Risk Management

# Appendices

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRSs overall set of requirements.

Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.