

Centralized Traffic Control Graphical User Interface Software Requirements Specification

Version 1.0

February 9, 2012

PAAC Demonstration System

|  |  |  |
| --- | --- | --- |
| List Of Revisions | | |
| Date | Name | Description |
| 2/9/12 | Jeremy Nelson | Initial version |

# Table of Contents

[List Of Revisions 2](#_Toc316501852)

[Table of Contents 2](#_Toc316501853)

[1 Introduction 2](#_Toc316501854)

[1.1 Purpose 2](#_Toc316501855)

[1.2 Scope 2](#_Toc316501856)

[1.3 Definitions and Abbreviations 2](#_Toc316501857)

[2 Risk Assessment 2](#_Toc316501858)

[2.1 Simulation Information 2](#_Toc316501859)

[2.2 Command Interface 2](#_Toc316501860)

# Introduction

## Purpose

The purpose of this document is to evaluate and assess potential risks associated with the successful completion of the Centralized Traffic Control (CTC) Graphical User Interface (GUI) for the PAAC transit control simulation system.

## Scope

This document assesses potential risks associated with the CTC GUI component of the PAAC transit control simulation system. It does not directly include risks associated with the other subcomponents of the system.

## Definitions and Abbreviations

**CTC** – Centralized Traffic Control

**GUI** – Graphical User Interface

**UML** – Universal Modeling Language

# Risk Assessment

## Scheduling

The CTC GUI requires a heavy amount of visual design and coding. There is a risk of controls not being fully or cleanly implemented in the allotted time, resulting in possible bugs in the GUI or scheduling delays. Since the GUI is the “master” of the transit control simulation, this presents a medium risk if the component does not function properly. This module is assigned to a programmer with two years of C# GUI implementation to mitigate this risk. Additionally, the programmer will make every effort to meet deadlines defined in the CTC GUI work schedule to meet the final project target.

## Technology and Resource Risks

The transit control system will be implemented in C# using Microsoft Visual Studio 2010. In addition to coding, Bazinga plans to use Visual Studio for UML diagrams for documentation and for unit testing. Since the Bazinga programmers do not have experience in utilizing the latter purposes, this presents the need for extra time to learn the tools, which may impact the system work schedule. Additionally, this presents a risk of creating incorrect or confusing UML diagrams, which may result in flawed design communication. This risk will be mitigated by holding regular team meetings and frequent emails to communicate designs. Incomplete or unreliable unit testing may result in some software bugs. This risk will be mitigated by system level testing which should catch most noticeable bugs.

Bazinga will use GitHub for source control and defect tracking. This is a free online tool that is not flawless. The use of GitHub presents the risk of losing or corrupting source code. Verification by the code author that the upload was successful after every check in will mitigate this risk. The use of GitHub also may cause delays in uploading source code due to software struggles or server outages. This risk can be mitigated by a secondary code repository using a reliable tool such as Dropbox.