**Software Requirement Specifications**

Authors: Kyle Legters, Nick Faughey, Pavan Kottapalli, Korey Klinger, Alex Pickering

**Table of Contents**

1. Introduction
   1. Purpose
   2. Scope
   3. Definitions, acronyms, and abbreviations
   4. References
   5. Overview
2. Overall Description
   1. Product perspective
   2. Product functions
   3. User characteristics
   4. Constraints
   5. Assumptions and dependencies
3. Specific Requirements
   1. External Interfaces
   2. Functions
      1. User Class 1: CTC Office Manager
      2. User Class 2: Train Driver
      3. User Class 3: Train Engineer
   3. Performance Requirements
   4. Logical Database Requirements
   5. Design Constraints
   6. Software System Attributes
      1. Reliability
      2. Availability
      3. Security
      4. Maintainability
      5. Portability
      6. Organization of Requirements

**1 Introduction**

**1.A Purpose**

The purpose of this SRS is to lay out a set of requirements, functions, and features for the Centralized Traffic Control Center and Signaling System for the North Shore Extension of the Pittsburgh Light Rail Transit system.

The intended audience of this document is the Port Authority of Allegheny County.

**1.B Scope**

The software this project will produce includes the Centralized Traffic Control (CTC) Center and Signaling System.

The project will include the following five systems:

Track Model: Simulates the actual track of the North Shore extension.

Train Model: Simulates the physics behind a Flexity 2 Tram moving across the track.

Train Controller: Safety Critical component that controls the movement of the train.

Track Controller: Safety Critical component that controls the track and reports back to the CTC office.

CTC Office: Allows a dispatcher to schedule, dispatch, and monitor trains.

The project will have an automatic mode with preset scenarios to demo the systems listed above, and this demo will be capable of running faster than wall clock time.

Implementing the North Shore extension of the Pittsburgh Light Rail system will improve the status of public transportation throughout the city of Pittsburgh. Public transportation is important, as it provides jobs, minimizes traffic, and greatly reduces a city’s carbon footprint.

**1.C Definitions, acronyms, and abbreviations**

* **SRS** – Software Requirements Specifications
* **CTC** – Centralized Traffic Control
* **Carbon Footprint** – the amount of carbon dioxide and other carbon compounds emitted
* **System** – refers to the software product
* **Failure State/Mode** – any sort of failure that might occur in the system (broken rail, engine malfunction, brake malfunction, etc.)
* **JAR Files**  - a package file format that includes many Java Class files in one file
* **Java** – the programming language that will be used to build the software
* **Backward Compatible**  - can function if given an input intended for an older version
* .**csv file** – Comma Separated Value file – allows data to be saved in table structured format

**1.D References**

American Public Transportation Association

<http://www.apta.com/mediacenter/ptbenefits/Pages/default.aspx>

IEEE Recommended Practice for Software Requirement Specifications

<https://courseweb.pitt.edu/bbcswebdav/pid-20285500-dt-content-rid-7299133_1/users/jap182/Labs/Lab%201/IEEE%20830.pdf>

**1.E Overview**

Section 2: Provides a background for requirements to be detailed later in the document.

Section 3: Provides detailed requirements for the system that enable designers to design the

system accordingly, and testers to test the system accordingly.

2 Overall Description

**2.A Product Perspective**

This is an independent and totally self-contained product.

**2.B Product Functions**

This product shall provide functions for scheduling, dispatching, and viewing status of trains in the network. It will shall a user to accelerate or decelerate a train, change the status of the lights, doors, and temperature, and view the failure mode of any train. The product shall allow a user to at any time know of broken rails, shut down sections of track, or switch a portion of the track.

**2.C User Characteristics**

1. The intended user of the system has a general knowledge of the physics that allows a train to move across the track.
2. The user shall be able to calculate when a train should begin slowing down so that it will remain within the given speed limit for a given section of track.
3. The user should possess the training to accurately dispatch trains without risking two trains moving towards each other on the same section of track.
4. The user should also possess the knowledge and training the switch the tracks accordingly so that the trains remain in a safe state.
5. The user should have the intuition to recognize where all trains are on the track at all times, to ensure the safety of all passengers.
6. The user should possess the training and experience to know what to do if any part of the system goes into a failure state.

**2.D Constraints**

1. The system is a safety critical piece of software that can put the lives of hundreds of people at risk should it fail.
2. The system must be developed in such a way that should the user make an error in his/her input, the system will immediately recognize the user’s error and override the input before anything critical can happen.
3. The system must be adaptable to the layout of the track and the schedule of the trains.
4. The system must be developed in a way that any can be applied to any track, and operate at full functionality. The track should be an input, and not internal data.
5. The train schedule must also be adaptable. Each day trains will need to run at different times, and therefore the system must be able to run the trains in a different manner each day.
6. It is anticipated that failures will eventually occur. In the event that one of these failures does occur, the system must be capable of responding to the failure in a way that will allow the rest of the system to continue running while the failure is dealt with.

**2.E Assumptions and Dependencies**

This system is deployed and run with the assumption that the target computer is a Windows machine able to execute and run JAR files. If the target computer’s specifications or capabilities change, this document will be revisited and revised to reflect the changes.

3 Specific Requirements

**3.A External Interfaces**

The software shall accept input from users either in a train or in a central office.

**3.B Functions**

**3.B.A User Class 1: CTC Office Manager**

1. The system shall enable a central office manager to create train schedules
2. The system shall provide methods for dispatching individual trains to specific stations
3. The system shall provide real-time feedback on train locations and speeds
4. The system shall notify managers of unsafe situations immediately
5. The system shall allow the dispatcher to close sections of track for maintenance, and reopen when ready
6. The system shall allow the dispatcher to set the authority for a select train
7. The system shall allow the dispatcher to display the current state of the entire system

**3.B.B User Class 2: Train Driver**

1. The system shall allow an engineer to accelerate and decelerate his/her train
2. The system shall allow an engineer to view the status of all train systems
3. The system shall show the state of train doors and lights at all times
4. The system shall allow an engineer to open/close doors and turn lights on/off
5. The system shall provide an emergency braking function, which can be triggered by passengers
6. The system shall allow an engineer to increase/decrease the temperature of the train

**3.B.C User Class 3: Train Engineer**

1. The system shall notify the engineer immediately of any broken or damaged rails
2. The system shall notify the engineer of the location of a train on the track
3. The system shall allow the engineer to switch the track at any time
4. The system shall allow the engineer to control the lights on the track

**3.C Performance Requirements**

1. 90% of trains routed through the network shall arrive within 1 minute of scheduled time
2. 95% of internal commands shall be executed within 1 second
3. 95% of status updates shall be performed in under 1 second
4. The system shall not allow more than 1 user per computer
5. 90% of user input commands shall be executed within 2 seconds
6. The system shall not be inoperable or in a frozen state for more than 1 second
7. The system shall be capable of running faster than wall clock time, with the constraint that timing-specific calculations are accurate within 0.1% at extreme processing speeds

**3.D Logical Database Requirements**

External .csv files shall be used when structured data needs to be provided as an input, but no external database will be used for normal data storage. All data shall be stored in the system’s main memory, and accessed when needed. Module communication will be handled natively within Java.

**3.E Design Constraints**

Not using an external database may cause information to be accessed simultaneously, so data locks will be implemented to prevent data races and conflicts.

**3.F Software System Attributes**

**3.F.A Reliability**

The software shall achieve 100% reliability by handling all errors gracefully, and continuing to operate safely afterwards.

**3.F.B Availability**

The software shall be immediately available for use after starting, and shall continue to be available until shut down by a user.

**3.F.C Security**

No extreme security precautions are necessary, since this system will run on computers in a physically secure office.

**3.F.D Maintainability**

The software’s modules can be updated independently and remain backward-compatible, to allow for updates as needed.

**3.F.E Portability**

The software shall not be portable to any other system besides those described in this document to ensure reliability.

**3.F.F Organization of Requirements**

The system is presented as one inseparable module with only one mode of operation. There are three classes of users – CTC office workers, train engineers, and train drivers. 3.B is organized by user class.