Train Trax: Train Monitor for Positive Train Control Test Beds

Software Design Specification

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author |
| 1.0 | 10/25/2015 | Initial Version. Created temporary template for software design specification. | Stephen Jalbert  Rashad Madyun  Corey Sanders |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[Revision History](#h.gjdgxs)

[**1. Introduction**](#h.1fob9te)

[1.1 Purpose of this document](#h.3znysh7)

[1.2 Scope of the development project](#h.2et92p0)

[1.3 Definitions, acronyms, and abbreviations](#h.tyjcwt)

[1.4 References](#h.3dy6vkm)

[1.5 Overview of document](#h.1t3h5sf)

[**2. System architecture description**](#h.4d34og8)

[2.1 Overview of modules / components](#h.2s8eyo1)

[2.2 Structure and relationships](#h.17dp8vu)

[2.3 User interface issues](#h.3rdcrjn)

[**3. Detailed description of components**](#h.26in1rg)

[3.1 Component template description](#h.lnxbz9)

[3.2 Motion Detection Unit](#h.35nkun2)

[3.3 Train Navigation Library](#h.1ksv4uv)

[3.4 Train Position Database](#h.44sinio)

[3.5 Train Monitor Terminal GUI](#h.2jxsxqh)

[**4.0 Reuse and relationships to other products**](#h.z337ya)

[**5.0 Design decisions and tradeoffs**](#h.3j2qqm3)

[**6.0 Pseudo code for components**](#h.1y810tw)

[**7.0 Appendices**](#h.4i7ojhp)

# **1. Introduction**

## 1.1 Purpose of this document

The computer engineering department owns a Positive Train Control Test Bed that is intended to mirror a typical train environment. The purpose of the train track is to be a teaching tool for instructing students on creating safety critical software. It is desired for the department Positive Train Control Test Bed to be able to track the location in for each train for this reason. Like subway trains, the department Positive Train Control Test Bed is completely indoors, so a Global Position System (GPS) is not possible.

The purpose of this document is to describe the design for the Train Trax Train Monitor to assist the department with tracking trains as they move along the Positive Train Control Test Bed. It will cover the design for both the desktop application and the embedded system software.

## 1.2 Scope of the development project

Train Trax is estimate the position of each train operating along the Positive Train Control Test Bed accurately enough to allow Train Operators schedule trains to run close enough to operation on the same section of track with minimal risk of collision. Additionally, Train Trax should provide a means for Train Operators to easily control switches on the train track without the need to using any additional train control software. Train Trax is only a monitor for trains, not train controller software. Train Trax consists of hardware that is equipped onto either the train engine or rail cars to measure train movement, software that will run on existing equipment within the department to graphically display train positions and to control movement. Furthermore, the development team is to assist the department with any modifications necessary to the Positive Train Control Test Bed to support proper operation of Train Trax, including the placement of markers on the track at pre-designated locations.

## 1.3 Definitions, acronyms, and abbreviations

To be added

## 1.4 References

IEEE Standard 1016: Software Design Specification

## 1.5 Overview of document

The remainder of the SDS will provide an overview of the system architecture and then describe the detailed design of each of the system components.

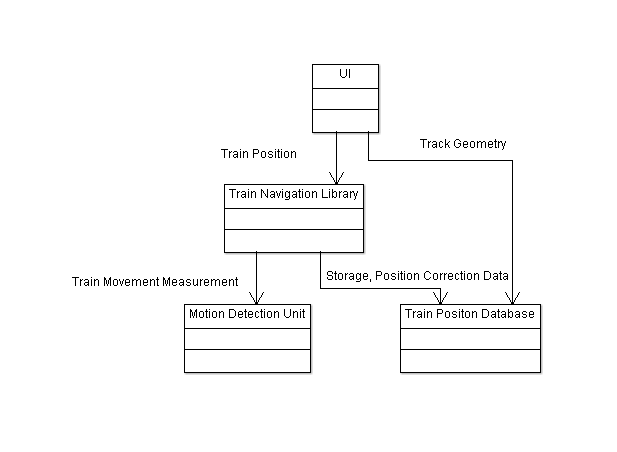
# **2. System architecture description**

## 2.1 Overview of modules / components

The Train Trax project consists of four top level components: the Motion Detection Unit used to collect position data, the Train Navigation Library used to calculate the position of trains with the raw data collected from the Motion Detection Unit, the Train Navigation Database used to store position data for the track and trains, and the Train Monitor Terminal GUI which will display train position and other system data to the user.

## 2.2 Structure and relationships

The Train Trax project contains a number of interrelations and dependencies. The Motion Detection unit is related to and depended on by the Train Navigation Library to provide the raw data for the navigation calculations for each train. There will be a relationship between the Train Navigation Library and the Train Position Database, where the Train Navigation Library will store processed position data for each train in the database. The Train Navigation Library depends on the Train Monitor Terminal GUI, which provides controls for system operation. The Train Monitor Terminal GUI depends on the Train Position Database to provide the geometry data for the track and position data for each train.



**Figure 1 Train Trax system relationships**

Make clear the interrelationships and dependencies among the various components. Structure charts can be useful here. A simple finite state machine can be useful in demonstrating the operation of the product. Include explanatory text to help the reader understand any charts.

## 2.3 User interface issues

To be added

This section will present the main principles of the product's user interface. Use the personas defined in section 2.1 of your SRS to make specific examples. This section should not touch on technical details. You may want to include sketches and specific text messages.

# **3. Detailed description of components**

## 3.1 Component template description

This section is not part of your design. It is the pattern you will use to describe the components given in subsections 3.2 - 3.n. Each part of the template will be identified by a label. Here in 3.1, you must briefly explain the purpose of each point. To make the presentation clear, use a table or bullet list. You may adapt the template suggested below to your particular needs (although deviations from the suggested template should be minimal and well-motivated).

## 3.2 Motion Detection Unit

## 3.3 Train Navigation Library

## 3.4 Train Position Database

## 3.5 Train Monitor Terminal GUI

These sections should contain design information for each component. Table included after appendices details contents.

# **4.0 Reuse and relationships to other products**

All software for the Train Trax project will be new development and no reuse will occur.

# **5.0 Design decisions and tradeoffs**

Use this section to motivate any decisions that will help the reader understand the design that your team is using. This section can also capture good ideas that were abandoned and the reasons for leaving them out of the design.

# **6.0 Pseudo code for components**

To be added as developed

# **7.0 Appendices**

To be added as necessary

Content guide for section 3.X (Table to be removed on document completion)

|  |  |
| --- | --- |
| Identification | The unique name for the component and the location of the  component in the system. |
| Type | A module, a subprogram, a data file, a control procedure, a class, etc |
| Purpose | Function and performance requirements implemented by the design component, including derived requirements. Derived requirements are not explicitly stated in the SRS, but are implied or adjunct to formally stated SDS requirements. |
| Function | What the component does, the transformation process, the specific inputs that are processed, the algorithms that are used, the outputs that are produced, where the data items are stored, and which data items are modified. |
| Subordinates | The internal structure of the component, the constituents of the component, and the functional requirements satisfied by each part. |
| Dependencies | How the component's function and performance relate to other  components. How this component is used by other components. The other components that use this component. Interaction details such as timing, interaction conditions (such as order of execution and data sharing), and responsibility for creation, duplication, use, storage, and elimination of components. |
| Interfaces | Detailed descriptions of all external and internal interfaces as well as of any mechanisms for communicating through messages, parameters, or common data areas. All error messages and error codes should be identified. All screen formats, interactive messages, and other user interface components (originally defined in the SRS) should be given here. |
| Resources | A complete description of all resources (hardware or software) external to the component but required to carry out its functions. Some examples are CPU execution time, memory (primary, secondary, or archival), buffers, I/O channels, plotters, printers, math libraries, hardware registers, interrupt structures, and system services. |
| Processing | The full description of the functions presented in the Function subsection. Pseudocode can be used to document algorithms, equations, and logic. |
| Data | For the data internal to the component, describes the representation method, initial values, use, semantics, and format. This information will probably be recorded in the data dictionary. |