Software Requirements Specification

Distributed Fence Vibration Monitor System

Version 1.1

Prepared by

J. Ramsay

B. Braithwaite

R. Mena

M. Paff

Embry-Riddle Aeronautical University

Table of Contents

[1 Introduction 3](#_Toc126151524)

[1.1 Purpose 3](#_Toc126151525)

[1.2 Document Conventions 3](#_Toc126151526)

[1.3 Intended Audience and Reading Suggestions 3](#_Toc126151527)

[1.4 Programming Environment 3](#_Toc126151528)

[1.5 Dependency Tree 4](#_Toc126151529)

[2 System Features (Functional Requirements) 5](#_Toc126151530)

[2.1 Motion Detection 5](#_Toc126151531)

[2.2 Communication 5](#_Toc126151532)

[2.3 Individual Nodes 5](#_Toc126151533)

[2.4 Central Server & Reporting Dashboard 7](#_Toc126151534)

[3 Non-Functional Requirements 8](#_Toc126151535)

[3.1 Performance Requirements 8](#_Toc126151536)

[3.2 Safety Requirements 8](#_Toc126151537)

[3.3 Security Requirements 8](#_Toc126151538)

[3.4 Software Quality Attributes 8](#_Toc126151539)

[4 Analysis Models 9](#_Toc126151540)

[5 Glossary 11](#_Toc126151541)

[6 References 12](#_Toc126151542)

# Introduction

## Purpose

The purpose of the Software Requirement Specification is to provide documentation for the software aspects of the Distributed Fence Vibration System. This document provides both non-functional and functional requirements. It is intended to be used to reference the functionality and design of the system within the scope of hardware and software. The document will present software specifications for the individual nodes and its components and the central server as a secondary requirement.

## Document Conventions

Refer to [CML for style guide] for reference to design.

## Intended Audience and Reading Suggestions

This Software Requirements Specification (SRS) document is primarily intended for the engineering team and design leads. It is suggested that this document be read with and referenced with the Hardware Requirement Specifications.

## Programming Environment

The components of the system shall be programmed using Visual Studio Code version 1.75. The MKR1310 as well as the RP2040 will be independent of each other with environments set up within an extension called Platform IO that is supported by Visual Studio Code. Platform IO is an extension that will allow for building and uploading of Arduino libraries and scripts for the individual boards. The Arduino mbed OS Nano board manager and the Arduino SAMD 32-bit Cortex M0+ will be installed from the Arduino IDE but supported and managed by Platform IO within Visual Studio Code. The files for each board consists of a header file and a source code file. Additional build flags in a configuration file enable advanced setup.

## Dependency Tree

* Source Files
  + dfm\_mkr1310.cpp
    - dfm\_mkr1310.h
    - adafruit\_ADXL345\_U.h
    - Arduino.h
      * ArduinoCore API for ….
    - ArduinoECCX08.h
    - ArduinoLowPower.h
    - LoRa.h
      * SPI.h
    - RTCZero.h
      * SPI.h
  + dfm\_nanorp2040.cpp
    - Arduino.h
    - Arduino\_LSM6DSOX.h
    - WiFiNINA.h
  + dfm\_vecv\_mkr1310.cpp
    - Arduino.h
    - ARduinoECCX08.h
    - LoRa.h
    - SPI.h
  + dfm\_recv\_nanorp2040.cpp
    - Arduino.h
    - WiFiNINA.h
  + main.cpp
    - Arduino.h
    - dfm\_vecv\_mkr1310.h
    - dfm\_nanorp2040.h

# System Features (Functional Requirements)

## Motion Detection

The software shall include a feature that detects motion from the chosen IMU and convert that motion into readable and transcribe data to then be used for other requirements. The software will include and depend on the libraries given from the chosen board and the components. This system feature holds the highest priority of all other software features as it is the core fundamental aspect of the project. The motion detection shall be interpreted and managed by the MKR1310 (or RP2040).

## Communication

Communication between each of the nodes and the central server will be done by LoRa and a created struct contained within the dfm\_mkr1310.h file will be transmitted using LoRa. This struct contains the information of the nodes. Each node will be in sleep mode until a hardware ISR occurs from the ADXL345 or another IMU and once the node is awake, after a synchronous clock trigger occurs, the awaken node with send a signal to other nodes to perform a state transition from sleep to awake.

## Individual Nodes

Each individual node will need to be able to communicate with other nodes and be able to communicate within a central server. The nodes will be able to manage and control their power consumption and status.

1. A node will communicate with other nearby nodes as part of a larger network.
   1. A node will attempt to use LoRa to communicate with a central server, or hub.
   2. A node out of range of the hub may use LoRa to communicate with other nodes.
2. A node will have two modes: Active and Sleep.
   1. A node will switch to Active mode when a hardware interrupt is received from a sensor.
      1. The sensor is an IMU with a fixed motion threshold.
   2. A node will analyze the vibration data and determine whether or not it is an attack.
   3. Whether the node determines that the vibration is an attack or not, it notifies the central server.
   4. A node will switch to Sleep mode when no interrupts occur for a configurable amount of time.
3. The system shall monitor vibrations for intrusions.
   1. The system shall detect vibrations via the accelerometer.
      1. Power density? Fourier Analysis? Thresholds?
      2. Rolling-buffer memory, FFT
      3. Have an EE help you here.
   2. When vibrations matching an intrusion pattern are detected, the node will send out an alert.
   3. Alerts are comprised of a packet containing a status.
      1. Status describes the severity of the attack.
4. The system shall report when a node has become disconnected.
   1. A disconnected node is a node that is no longer sending or receiving signals.
5. Each node shall periodically report diagnostics to the system.
   1. A node shall report its battery level as a percentage of its full charge.
   2. A node shall report its internal temperature in degrees Celsius.
   3. A node shall report its uptime, time spent awake, and time spent transmitting.
   4. A node shall report its configuration:
      1. Transmitting frequency and Sync Word (SW, similar to a channel)
      2. Epoch time as noted by its RTC
      3. Spread Factor (SF)
6. A node shall have an identifier that can be used to determine its relative location.
   1. Adjacent nodes shall have an ID that differs by 1.
7. A node shall communicate by broadcasting a packet.
   1. Nodes shall relay the packets sent by other nodes (if they are awake).
   2. Each packet sent shall have a unique ID.
      1. To avoid infinitely cascading signals, a node shall not relay subsequent signals received with an identical ID.
      2. When a node relays a signal, the node shall record the unique ID of that signal in a buffer.

## Central Server & Reporting Dashboard

Though not being fully designed and produced for at this time, the requirement for the central nodes consists of the fence nodes communicating with the central server. The central server will be for data processing outside of each individual node.

1. The server shall receive information from an arbitrary number of nodes.
2. The server shall record the data transmitted from the nodes.
   1. The record is transmitted via USB to a file on a computer.
   2. Data collected from received packets:
      1. Timestamp
      2. Packet author
      3. Sequence number
      4. Status
      5. Battery level
      6. Configuration information
3. The server shall recognize when a node needs attention or repair.
   1. **[NTH]** Notify personnel.
4. **[NTH]** The system shall correct the real-time clock value of improperly set nodes.

# Non-Functional Requirements

## Performance Requirements

1. **[NTH]** The system shall dynamically adjust power consumption to meet production in order to operate indefinitely.

## Safety Requirements

1. The software must not cause damage to the hardware.
   1. (I mean it’s pretty hard to do on accident but good idea nonetheless)
2. All software shall be provably able to run indefinitely without encountering critical errors.

## Security Requirements

1. **[NTH]** The system shall implement time-varying rolling-codes to prevent a replay attack.
2. **[NTH]** The software will utilize the ATECC508 cryptographic co-processor to ensure that the data that is being transmitted is secure. ///////Remove this!!!!!!!!!!!
   1. The system shall utilize a fixed packet structure to format data received from the nodes.
   2. The system shall implement time-varying rolling-codes to prevent a replay attack.
   3. The system shall intermittently send obfuscating data to prevent pattern recognition attacks.
3. **[NTH]** Very old packet IDs can be ignored or reported as an attempted replay attack.
4. **[NTH]** Intentional run-down detection.

## Software Quality Attributes

1. The software will be able to run indefinitely without encountering serious errors.
2. The software will be able to run smoothly on the specified processor.
3. The actual code shall be neatly maintained.
   1. GitHub synchronization
   2. Automatic Formatting
      1. K&R Indentation
4. All libraries must be open source.
   1. MIT or LGPL Licenses
   2. Libraries shall not be modified

# Analysis Models

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UC-IN-1 | | |
| Use Case Name: | Individual Nodes - Communication | | |
| Created By: | Renzo Mena | Last Updated By: | Renzo Mena |
| Date Created: | 1/23/2023 | Date Last Updated: | 1/24/2023 |

|  |  |
| --- | --- |
| Actor: | The System/ Individual node |
| Description: | Once the system is linked to the network, it should be able to connect to other nearby nodes. After the link is established, the system should be able to communicate and inform the network of discrepancies (i.e., disconnected nodes, the system being under attack, and vibrations due to intrusions). Along with abnormalities the system will report its own status (i.e., diagnostics, battery level, and location relative to other nodes) |
| Preconditions: | The system is setup, configured, and linked to the network. |
| Postconditions: | The system reports any abnormalities and basic information to the server and other nodes |
| Priority: | High |
| Frequency of Use: | Routine |
| Normal Course of Events: | 1. System runs diagnostics and report to the server. 2. Battery life is relayed to the server. 3. Vibration monitoring |
| Alternative Courses: | 1. Disconnected node is detected and is reported. 2. System attack is detected and is reported. 3. Intrusion is detected and is reported. |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | The system was successfully configured and is in working condition. |
| Notes and Issues: | None |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UC-CSRD-1 | | |
| Use Case Name: | Central Server & Reporting Dashboard | | |
| Created By: | Renzo Mena | Last Updated By: | Renzo Mena |
| Date Created: | 1/23/2023 | Date Last Updated: | 1/24/2023 |

|  |  |
| --- | --- |
| Actor: | The server |
| Description: | The server receives and records data transmitted from multiple nodes. Also, if the server receives an alert for repair or damages from a node, personnel will be contacted and sent to the location |
| Preconditions: | A links to the server and nodes are established via LoRaWAN |
| Postconditions: | The server continuously receives information and report to personnel |
| Priority: | High |
| Frequency of Use: | Constant |
| Normal Course of Events: | 1. The server records information 2. Personnel is informed and sent to location |
| Alternative Courses: | None |
| Exceptions: | None |
| Includes: | Individual nodes |
| Special Requirements: | None |
| Assumptions: | Server and nodes are properly configured, and the connection is stable |
| Notes and Issues: | None |

# Glossary

The below table contains acronyms and abbreviations used in this document.

|  |  |
| --- | --- |
| Abbreviation | Description |
| ID | Identifier |
| IDE | Integrated Development Environment |
| IMU | Inertial Measurement Unit |
| IO | Input/Output |
| LoRa | Long Range proprietary radio communication technique |
| *LoRaWAN* | IoT, Low power Wide Area Networking protocol |
| NTH | Nice To Have |
| OS | Operating System |
| RTC | Real Time Clock |
| SRS | Software Requirements Specification |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Revision History

|  |  |  |
| --- | --- | --- |
| Date | Revision | Changes |
| 1/25/2023 | 1.0 | First Release |
| 2/??/2023 | 1.1 | (changes after Jaffe reviews it) |

# References