A real-time multi-channel displacement monitoring and control software for a 17-channel SMA actuator system.

Software Requirements Specification

Version 1.0

23/01/2026

Group K

Prepared for
CS 310 Software Engineering
Spring 2026

Revision History

Date	Description	Author	Comments
23/01/26	Version 1	Group K	All the basic requirements from client covered

Document Approval

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

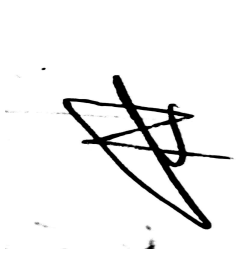
Signature	Printed Name	Title	Date
	Nikunj Bheda	A real-time multi-channel displacement monitoring and control software for a 17-channel SMA actuator system	23/01/26

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to describe the functional and non-functional requirements of the real-time multi-channel displacement monitoring and control software for a 17-channel Shape Memory Alloy (SMA) actuator system. This document is intended for project developers, evaluators and customers.

1.2 Scope

The software product to be developed is a windows desktop application.
The system will allow the user to:

- Control and monitor each SMA actuator individually in open and closed loop modes.
- Input desired displacement values for each SMA actuator.
- Automatically compute the required voltage or control signal.
- Send control commands to the hardware controller (Arduino).
- Monitor real-time displacement of the SMA actuators.
- Visualize data using graphs and tables.
- Modify the equation representing the relationship between voltage and displacement.

The system will not handle low-level hardware circuit design. It assumes that appropriate sensors and circuits are already available.

1.3 Definitions, Acronyms, and Abbreviations

- SMA – Shape Memory Alloy
- GUI – Graphical User Interface
- MCU – Microcontroller Unit

1.4 Overview

This document is organized into three main sections.

- Section 1 introduces the purpose and scope of the project.
- Section 2 provides a general description of the system and its users.
- Section 3 specifies detailed functional and non-functional requirements of the software

2. General Description

2.1 Product Perspective

The proposed product is a windows desktop application that interacts with an external microcontroller-based SMA hardware setup. The software as a logic layer to the microcontroller that handles the hardware.

2.2 Product Functions

The software will allow users to:

- Configure the number of SMA actuators.
- Configure the equation driving the software.
- Work in open and closed loop modes.
- Input voltage values in the open loop mode.
- Input the desired displacement values for each actuator in closed loop mode and convert displacement to voltage.
- Transmit control signals to hardware.
- Stop the supply of voltage for each actuator individually.
- Monitor real-time displacements (in closed loop mode).
- Display real-time graphs.

2.3 User Characteristics

The intended users are researchers working on mechatronics. The users are expected to have a basic knowledge about electronics. They must also be familiar with operating a windows desktop.

2.4 General Constraints

- The system must run on Windows OS.
- Communication with hardware will be via USB serial interface.
- Real-time performance depends on microcontroller speed.

2.5 Assumptions and Dependencies

- Arduino or similar microcontroller is available.
- SMA driver circuit is properly designed.
- USB communication is stable

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The system shall provide a graphical user interface (GUI) with:

- Input fields for displacement and voltage values.
- Buttons for start, stop, and reset.
- Real-time plots of displacement vs time.
- Text fields for display of sensor data.

3.1.2 Hardware Interfaces

The software will interface with an Arduino microcontroller through a USB serial port.
The microcontroller shall be connected to:

- SMA actuators.

3.1.3 Software Interfaces

The system shall run on Windows and will use:

- C# and WPF for GUI.
- Arduino language for the Arduino code.
- Serial communication libraries.

3.1.4 Communications Interfaces

Communication between software and microcontroller will be over USB.

3.2 Functional Requirements

3.2.1 Manual (Open-Loop) Control Mode

3.2.1.1 Introduction

The manual (open-loop) control mode allows the user to directly specify the voltage for each of the SMA actuator channels.

3.2.1.2 Inputs

- Voltage values for each of the SMA actuator channels.
- Stop button to stop the supply of voltage to the controller.

3.2.1.3 Processing

- The system shall transmit the user-defined voltage values to the hardware controller.

3.2.1.4 Outputs

- The system shall output the amplified voltage for each actuator and whether the voltage has been successfully transmitted to the controller.

3.2.1.5 Error Handling

- If any of the voltage values is out of the valid range, the system shall return an error message.

3.2.2 Automatic (Closed-loop) Control Mode

3.2.2.1 Introduction

The automatic (closed-loop) control mode allows the user to specify desired displacement values, and the system automatically computes the actuator voltage to achieve the target displacement.

3.2.2.2 Inputs

- User-defined target displacement values for each channel.
- Stop button to stop the supply of voltage to the controller.

3.2.2.3 Processing

- The system shall compute the required voltage values for the target displacement for each actuator.

- The system shall transmit the voltage values to the hardware controller.
- The system shall compute the real-time displacement values of each actuator.

3.2.2.4 Outputs
- The system shall output the amplified voltage values required for the target displacement.
- The system shall output whether the voltage has been successfully transmitted to the controller.
- The system shall output the real-time displacement values of each actuator along with a graphical display of the same.

3.2.2.5 Error Handling
- If the user enters an invalid equation, the system shall display an error message.
- If the number of actuators exceeds supported limits, the system shall reject the configuration.

3.2.3 Configuration Module

3.2.3.1 Introduction
The configuration module allows the user to define system parameters such as the number of actuators and the mathematical relationship between actuator voltage and displacement.

3.2.3.2 Inputs
- User-defined number of actuator channels.
- User-defined calibration equation relating voltage to displacement.
- User-defined amplifier gain.

3.2.3.3 Processing
- The system shall store the configured number of actuators.
- The system shall apply the user-defined calibration equation for displacement computation.

3.2.3.4 Outputs
- Updated system configuration parameters.

3.2.3.5 Error Handling
- If the user enters an invalid equation, the system shall display an error message.
- If the number of actuators exceeds supported limits, the system shall reject the configuration.

…

3.3 Classes / Objects

3.3.1 UserClass

3.3.1.1 Attributes

- actuatorCount
- calibrationEquation
- amplifierGain

3.3.1.2 Functions

- setCalibrationEquation()
- setActuatorCount()
- setAmplifierGain()

3.3.2 ActuatorController Class

3.3.2.1 Attributes

- actuatorID
- voltageValue
- targetDisplacement

3.3.2.2 Functions

- setVoltage()
- computeRealTimeDisplacement()
- computeVoltage()
- sendControlSignal()

3.4 Non-Functional Requirements

3.4.1 Performance

- In open-loop mode, the supplied voltage shall be computed directly using equations provided by the authenticated user, resulting in a computation delay of **less than 1 ms**.
- In closed-loop mode, the desired displacement shall be provided as input, and the required voltage for each control step shall be computed within **less than 1 ms** per time interval dt, ensuring computation time remains smaller than dt. These computations shall be executed iteratively until the desired displacement is achieved for the corresponding channel.
- Use of pre-supplied, validated equations shall minimize software-induced latency.

3.4.2 Reliability

- Equations used for voltage computation shall be stored within the application and editable only by authenticated users. Modifications shall be permitted exclusively via an explicit **Save** action and shall be performed **atomically** (write to a temporary file and replace the original via atomic rename) so that, in the event of an application crash during update, the previously stored equations remain intact. The application shall validate

equations on load and before save for syntactic correctness and numerical stability; invalid equations shall be rejected and no change applied. All equation edits and save actions shall be recorded in an append-only audit log with timestamp and user identifier.

3.4.3 Security

- The software is protected by password authentication and only the correct user-password pair will be granted the access

3.4.4 Maintainability

- Modular architecture**.** Separate layers: hardware abstraction, control logic, UI, persistence.
- Developer and user manuals for understanding and changing purpose

3.4.5 Portability

- Windows 10/11, 64-bit. Apps must run on typical lab laptops.

3.5 Inverse Requirements

The software shall not

- Autonomously modify actuator control parameters without explicit user input.
- Apply voltage values exceeding predefined safe voltage limits for any SMA actuator.
- Require physical displacement sensors for operation in simulated closed-loop mode.
- Perform adaptive or self-learning control algorithms such as neural networks or reinforcement learning.
- Continue actuator operation after a global stop command is issued.
- Assume a fixed number of actuators, and shall not be hardcoded specifically for 17 channels.
- Store or transmit actuator data to external servers or cloud services.

3.6 Design Constraints

- The software shall be developed as a Windows executable (.exe) application.
- Voltage outputs shall be constrained by hardware-safe voltage limits defined by the SMA actuator specifications.
- Closed-loop operation shall initially rely on software-simulated feedback models rather than physical sensors.
- The user interface shall allow operation using standard keyboard and mouse input only.

3.7 Logical Database Requirements

Persistent storage shall not be mandatory for basic system operation.

For each actuator, the system shall store:

- Actuator ID
- Applied voltage
- Target displacement

3.8 Other Requirements

- The software shall provide real-time graphical plots of displacement versus time for each SMA actuator.
- The system shall allow users to apply global commands, including
    - Stop all actuators.
    - Set a common displacement for all actuators
    - Set a common voltage for all actuators.
- The system shall support adjustable amplification gain for control signal scaling.
- The user shall be able to enable or disable individual actuators at runtime.
- The system shall allow future integration of real displacement sensors without major architectural changes.
- The software shall display meaningful error and warning messages for invalid user inputs.

A. Appendices

The information contained in the appendices is provided for reference only and does not constitute mandatory software requirements.

A.1 Appendix 1 – SMA Actuator Modeling Assumptions

This appendix describes the assumptions made while modeling the Shape Memory Alloy (SMA) actuator behavior in software during simulated closed-loop operation.

Since real-time displacement sensors are not integrated in the current phase, the system uses a software-based SMA model to simulate actuator displacement as a function of applied voltage and time. The model assumes a monotonic relationship between applied voltage and actuator displacement.

These assumptions are used solely for software validation and user interface testing. Future versions of the system may replace the simulated model with real sensor feedback.

A.2 Appendix 2 – System Use Case Summary

This appendix summarizes typical use cases supported by the SMA actuator control software:

- **Open-loop Control:**
  The user directly applies voltage values to one or more SMA actuators without feedback control.

- **Closed-loop Control :**
  The user specifies a target displacement, and the system computes and applies the required voltage using a simulated feedback model.

- **Global Control Operations:**
  The user applies global commands such as stopping all actuators or setting common displacement values.

- **Monitoring and Visualization:**
  The user observes real-time displacement versus time graphs for selected actuators.

These use cases are referenced by the functional requirements defined in Section 3.2.