Software Requirements Specification
for
Odyssey Space Research

Software Engineer Evaluation

Version 0.1

Prepared by **Frank Putnam, Jr.**

Date of submission: **Friday, June 7, 2019**

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
0.1	Frank Putnam, Jr.	Initial draft	2019/06/07

1 Introduction

1.1 Document Purpose

The purpose of this document is to outline software specifications of a C++ project to evaluate my software engineering abilities. The customer requirements were deliberately vague in order to review my personal decisions related to the implementation.

1.2 Product Scope

The product is a C++ class implementing a 2 dimensional matrix of form n x m, where n and m are set by the user when the object is constructed. The class will support operations of addition, scalar multiplication, matrix multiplication, and transpose (my choice). The overloaded C++ operators of ***get*** and ***set*** will manipulate data of a specific element in the matrix using unit-based access.
Unit tests will be created by me and included in the zipped git repository submission.

1.3 Intended Audience

The product will be evaluated by at least two senior software developers from Odyssey Space Research.

1.4 Definitions, Acronyms and Abbreviations

<*Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.*
TO DO: Please provide a list of all abbreviations and acronyms used in this document sorted in alphabetical order.>

1.5 Document Conventions

In general this document follows the IEEE formatting requirements. Arial font size 11 or 12 is used throughout the document for text. *Italics* are used for comments and ***bold italics*** are used for code operators. Document text is single spaced and maintains 1” margins.

1.6 References and Acknowledgments

<*List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document.* >

2 Overall Description

2.1 Product Overview

<*Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. In this part, make sure to include a simple diagram that shows the major components of the overall system, subsystem interconnections, and external interface. In this section it is crucial that you will be creative and provide as much information as possible.*

TO DO: Provide at least one paragraph describing product perspective. Provide a general diagram that will illustrate how your product interacts with the environment and in what context it is being used. This is not a formal diagram, but rather something that is used to illustrate the product at a high level.>

2.2 Product Functionality

<*Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary is needed here. These can be at the level given in the project description.>*
TO DO: Provide a bulleted list of all the major functions of the system

1) C++ class implementing a 2 dimensional matrix of form is **n** x **m.**
2) **n** and **m** are set by the user when the object is constructed.
3) Operations of
    a. addition
    b. scalar multiplication
    c. matrix multiplication
    d. transpose
4) Overloaded operators
    a. ***get*** to obtain data from a specific element of the matrix.
    b. ***set*** to load data into a specific element of the matrix.
    c. Matrix element access is unit-based.

2.3 Design and Implementation Constraints

<*Describe any items or issues that will limit the options available to the developers. These might include: hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software). You can be creative here to some degree. At a minimum, you need to identify that you must use the COMET method for software design and the UML modeling language. Make sure you provide references for both.* >

1) Standard C++ libraries.
2) Microsoft Visual Studio 2019 Professional (trial version).
3) Microsoft Visual C++ Redistributable for Visual Studio 2019.

2.4 Assumptions and Dependencies

<*List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project.*
TO DO: Provide a short list of some major assumptions that might significantly affect your design.>

The inherently subjective evaluation will factor the following items into the analysis:

a) Does the code compile?
b) Does it do what it should?
c) Does it pass all unit tests?
d) Are the unit tests sufficient?
e) How readable and maintainable is the code?
f) Is the code robust enough to handle any edge cases?
g) Were good habits exhibited that would be expected of a professional at your level of expertise?
h) Were C++ best practices followed (or failed to be followed)?
i) What extra features to these bare requirements were added?
j) Is the code efficient?
k) What design and implementation decisions were made, and why?
l) Were relevant comments and other documentation added?

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

<*Describe the logical characteristics of each interface between the software product and the users. For your project, you only need to be concerned with the main thermostat (not the mobile app) and can use the graphic from the project description as the basis for your user interface..*
TO DO: Provide the graphic for the thermostat user interface and provide a basic description as to how users will interact (e.g. tough screen, menus, etc.).>

3.1.2 Hardware Interfaces

Development and testing was performed on a Hewlett Packard Pavilion series laptop containing

a) x64 processor - AMD A8-6410 APU (4 CPUs) with AMD Radeon R5 graphics at 2.00 GHz.
b) DRAM - 16.0 GB.
c) Hard drive - Samsung 860 EVO SSD 1.00 TB.
d) Microsoft Windows 64-bit operating system.
    1) Windows 10 Home edition
    2) Version 1809
    3) OS build 17763.503

The C++ matrix class should run on any hardware platform as long as it is compiled for that platform.

3.1.3 Software Interfaces

<*Describe the connections between this product and other specific software components (in your case, just the mobile app that can send commands).*>

3.2 Functional Requirements

< Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. This section is the direct continuation of section 2.2 where you have specified the general functional requirements. Here, you should list in detail the different product functions.

3.2.1 F1: The system shall …

3.2.2 <Functional Requirement or Feature #2>

…

3.3 Use Case Model

TO DO: Provide a use case diagram that will encapsulate the entire system and all actors.

3.3.1 Use Case #1 (use case name and unique identifier – e.g. U1)

TO DO: Provide a specification for each use case diagram

Author – Identify team member who wrote this use case

Purpose - What is the basic objective of the use-case. What is it trying to achieve?

Requirements Traceability – Identify all requirements traced to this use case

Priority - What is the priority. Low, Medium, High. Importance of this use case being completed and functioning properly when system is depolyed

Preconditions - Any condition that must be satisfied before the use case begins

Post conditions - The conditions that will be satisfied after the use case successfully completes

Actors – Actors (human, system, devices, etc.) that trigger the use case to execute or provide input to the use case

Extends – If this is an extension use case, identify which use case(s) it extends

Flow of Events

1. Basic Flow - flow of events normally executed in the use-case
2. Alternative Flow - a secondary flow of events due to infrequent conditions
3. Exceptions - Exceptions that may happen during the execution of the use case

Includes (other use case IDs)

Notes/Issues - Any relevant notes or issues that need to be resolved

3.3.2 Use Case #2

…

4 Other Non-functional Requirements

4.1 Performance Requirements

<*If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.*
TODO: Provide performance requirements based on the information you collected from the client/professor. For example, you can say "P1. The secondary heater will be engaged if the desired temperature is not reached within 10 seconds">

4.2 Safety and Security Requirements

<*Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied. Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements.*
TODO:

- *Provide safety/security requirements based on your interview with the client - again you may need to be somewhat creative here. At the least, you should have some security for the mobile connection.*

4.3 Software Quality Attributes

<*Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.*

TODO: Use subsections (e.g., 4.3.1 Reliability, 4.3.2 Adaptability, etc…) provide requirements related to the different software quality attributes. Base the information you include in these subsections on the material you have learned in the class. Make sure, that you do not just write "This software shall be maintainable…" Indicate how you plan to achieve it, & etc…Do not forget to include such attributes as the design for change (e.g. adapting for different sensors and

*heating/AC units, etc.). Please note that you need to include **at least** 2 quality attributes. You can Google for examples that may pertain to your system.>*

5 Other Requirements

*<This section is **Optional**. Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

Appendix A – Data Dictionary

<Data dictionary is used to track all the different variables, states and functional requirements that you described in your document. Make sure to include the complete list of all constants, state variables (and their possible states), inputs and outputs in a table. In the table, include the description of these items as well as all related operations and requirements.>

Appendix B – Peer Review Log

<Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist in determining the effort put forth to produce this document>