SE 3XA3: Software Requirements Specification
RD-B V2

Team #, Team Name
Jason Tsui tsuij8
Student 2 name and macid
Student 3 name and macid

November 9, 2018

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
Nov 9,2018	1.0	Document Creation, Use Hierarchy Between Modules, Introduction, Connection Between Requirements and Design
Date 2	1.1	Notes

1 Introduction

This a Module Guide document for the project RD-B V2 created group 31 of McMaster University SE3XA3 Fall 2018. This documents covers present module design decisions, module behavior, tracibility of module implementations, and anticipated changes to module design. This document is intended to be used as a guideline for module design and overall strucutre of the project.

2 Anticipated and Unlikely Changes

This section lists possible changes to the system. According to the likeliness of the change, the possible changes are classified into two categories. Anticipated changes are listed in Section 2.1, and unlikely changes are listed in Section 2.2.

2.1 Anticipated Changes

Anticipated changes are the source of the information that is to be hidden inside the modules. Ideally, changing one of the anticipated changes will only require changing the one module that hides the associated decision. The approach adapted here is called design for change.

AC1: The specific hardware on which the software is running.

AC2: The format of the initial input data.

...

2.2 Unlikely Changes

The module design should be as general as possible. However, a general system is more complex. Sometimes this complexity is not necessary. Fixing some design decisions at the system architecture stage can simplify the software design. If these decision should later need to be changed, then many parts of the design will potentially need to be modified. Hence, it is not intended that these decisions will be changed.

UC1: Input/Output devices (Input: File and/or Keyboard, Output: File, Memory, and/or Screen).

UC2: There will always be a source of input data external to the software.

...

3 Module Hierarchy

This section provides an overview of the module design. Modules are summarized in a hierarchy decomposed by secrets in Table 2. The modules listed below, which are leaves in the hierarchy tree, are the modules that will actually be implemented.

M1: Hardware-Hiding Module

...

Level 1	**Level 2**
Hardware-Hiding Module	
Behaviour-Hiding Module	? ? ? ? ? ? ? ?
Software Decision Module	? ? ?

Table 2: Module Hierarchy

4 Connection Between Requirements and Design

The system is decomposed into modules for information hiding and separated based on the requirements of the design in the SRS. Table 3 highlights the connection between the requirements and implemented modules.

5 Module Decomposition

Modules are decomposed according to the principle of "information hiding" proposed by Parnas et al. (1984).

5.1 Hardware Hiding Modules (M1)

Secrets: The data structure and algorithm used to implement the virtual hardware.

Services: This module serves as the interface between the hardware and software of the program. This is done automatically and abstracted by the operating system.

Implemented By: OS

5.2 Behaviour-Hiding Module

Secrets: The contents of the required behaviours.

Services: This module serves as the external interface between the system specified by the software requirements specification and the user. This module acts as a communication layer between the hardware-hiding module and the software decision module. This is done and abstracted by the Discord application and API.

Implemented By: Discord application and API

5.3 Input Format Module

Secrets: The format and structure of the input data.

Services: Converts the input data from the behaviour-hiding module into the data structure used by the input parameters module.

Implemented By: RD-B V2

5.4 Software Decision Module

Secrets: The design decision based on mathematical theorems, physical facts, or programming considerations. The secrets of this module are *not* described in the SRS.

Services: Includes data structure and algorithms used in the system that do not provide direct interaction with the user. Performs logical computation and returns output to behaviour-hiding module for output.

Implemented By: RD-B V2

6 Traceability Matrix

This section shows two traceability matrices: between the modules and the requirements and between the modules and the anticipated changes.

Req.	Modules
R1	M1, M??, M??, M??
R2	M??, M??
R3	M??
R4	M??, M??
R5	M??, M??, M??, M??, M??, M??
R6	M??, M??, M??, M??, M??, M??
R7	M??, M??, M??, M??, M??
R8	M??, M??, M??, M??, M??
R9	M??
R10	M??, M??, M??
R11	M??, M??, M??, M??

Table 3: Trace Between Requirements and Modules

AC	Modules
AC1	M1
AC2	M??
AC??	M??
AC??	M??
AC??	M??
AC??	M??
AC??	M??
AC??	M??
AC??	M??
AC??	M??
AC??	M??
AC??	M??

Table 4: Trace Between Anticipated Changes and Modules

7 Use Hierarchy Between Modules

Table 2 outlines the hierarchy between modules. Use hierarchy refers to modules requiring the correct function of another module in order to function correctly.

Figure 1: Use hierarchy among modules

References

D.L. Parnas, P.C. Clement, and D. M. Weiss. The modular structure of complex systems. In *International Conference on Software Engineering*, pages 408–419, 1984.