CS 461 - Fall 2016 - Design Document

Project DevAI

Jacob Broderick, Kristen Patterson, Brandon Chatham

Abstract

The goal of this project is to create an agent to play the game Starcraft Brood War, a real time strategy game created by Blizzard Entertainment. The solution to the project will be a template that future students can use to develop better solutions than those provided in this project. In this document, we will discuss how we will design and implement our project. The design choices we make will help as a road map and checklist to follow.

CONTENTS

I. Introduction

A. Date of issue and status

B. Scope

C. Issuing organization

D. Authorship

E. References

F. Context

G. Design languages

H. Body

I. Summary

J. Glossary

K. Change history

II. Design Stakeholders

A major stakeholder in the design process is future students involved in an Oregon State University club. These students will be reading and using the code included in the project.

III. Design concerns

The design concerns of the future students are that the code is well documented and organized.

IV. Design Views

The code is well documented if it has concise comments and includes functional header comments. In order for the code to be organized, it must be modularized into simple functions and further separated into header files containing functions with similar purposes. The functions also need to be listed in order of complexity with the most complex functions requiring the highest amount of dependencies is located at the bottom.

V. Design Viewpoints

A. Pattern

The pattern viewpoint covers the design for the commenting of the code. Commenting in the program will require reuse of function header comments to describe the function's purpose. Functional header comments will follow a specified template of describing the name, process, inputs, outputs, and requirements of a function. In-line comments must be included for any variable declarations, loops, and function calls as well as any ambiguous declarations. The parts of the functional header comment must directly associate with code in the function. For example, the inputs are the parameters of a function and the outputs are the returned values. This viewpoint will be supported using the UML composite structure diagram.

B. Composition

The composition viewpoint is used for the design of modular functions and files. Composition will help to localize and simplify blocks of code into functions. Functions will then be localized into separate files based on their purpose. Most functions will be decomposed into modules with a singular purpose while their will be a few functions that control the system and call upon the simpler modules. The HIPO diagram will be used to supplement this viewpoint.

C. Dependency

The dependency viewpoint designs the order of the functions in each file. The design of each file will follow the order of least complex function with least dependencies on top while more complex functions that use other functions are on the bottom of the file. A UML component diagram will be used to show the dependencies.

VI. Design Overlays

VII. Design Rationale

A. Commenting

Adding function header comments and using a template will help future students to quickly and easily find the information they need about the function. In-line comments will also help future students follow the process in the function.

B. Modular functions and files

Keeping functions modular and organized into separate files will help future students to easily navigate the code. It will also help them use the code as the can take separate modules or files easily.

C. Ordered functions

Ordering functions by complexity will help the future students navigate as they can quickly find simple functions to use. They will also be able to physically see the use of simple functions to support larger more complex functions.