ECSE 321 Introduction to Software Engineering

McGill University Winter 2015

Tower defense project:  
Software Design Specifications

Andrei Chubarau 260581375

Yordan Neshev 260587938

Dang Khoa Do 260584925

Steven Voyer 260531264

Table of Contents

[1 Introduction 2](#_Toc414197302)

[1.1 Purpose of the document 2](#_Toc414197303)

[1.2 audience 2](#_Toc414197304)

[1.3 Scope 2](#_Toc414197305)

[1.4 Related documents 2](#_Toc414197306)

[2 System Overview 3](#_Toc414197307)

[3 Design Considerations 4](#_Toc414197308)

[3.1 Assumptions and Dependencies 4](#_Toc414197309)

[3.2 Constraints 4](#_Toc414197310)

[3.3 Goals and Guidelines 4](#_Toc414197311)

[3.4 Development Methods 4](#_Toc414197312)

[4 System Architecture 5](#_Toc414197313)

[4.1 Architectural Strategies 5](#_Toc414197314)

[4.2 Architectural Diagram 5](#_Toc414197315)

[5 Detailed System Design 6](#_Toc414197316)

[5.1 Component level design 6](#_Toc414197317)

[5.1.1 Layer 1 6](#_Toc414197318)

[5.1.2 Layer 2 6](#_Toc414197319)

[5.1.3 Layer 3 6](#_Toc414197320)

[5.1.4 Layer … 6](#_Toc414197321)

[5.2 Object Oriented Principles and Patterns 7](#_Toc414197322)

[5.2.1 Principles Used 7](#_Toc414197323)

[5.2.2 Pattern Used 7](#_Toc414197324)

[6 Dynamic Behaviour: Sequence Diagrams 8](#_Toc414197325)

[7 (System Logic…) Design 9](#_Toc414197326)

[8 User Interface Design 10](#_Toc414197327)

[8.1 Description of the User Interface 10](#_Toc414197328)

[8.2 Visual Presentation of Graphical User Interface 10](#_Toc414197329)

[9 References 11](#_Toc414197330)

# Introduction

## Purpose of the document

## audience

## Scope

## Related documents

# System Overview

# Design Considerations

## Assumptions and Dependencies

## Constraints

## Goals and Guidelines

## Development Methods

# System Architecture

## Architectural Strategies

The design of the current system is based on the layered architectural style, where the main system is divided into two main subcomponents, namely Presentation and Business Logic layers. Such an approach limits interaction between the two layers and separates responsibilities to achieve higher system cohesion.

The previously mentioned duality of the system ensures separation of concerns. Business Logic layer handles all system parameters and user inputs, whereas Presentation layer has the responsibility of displaying the resulting system behaviour. This contributes to managing system complexity by separating system domain logic and system view functionality.

Moreover, Presentation layer depends on Business Logic layer as the latest provides services and information to the first. The Model-View separation principle is applied as Business Logic has no dependency on the Presentation layer. This design style contributes to high cohesion and low coupling of the system. Additionally, this allows the system to display the same domain logic component in different presentation styles, thus enhancing reuse and extendibility.

## Architectural Diagram

[INSERT DIAGRAM HERE]

# Detailed System Design

## Component level design

System subcomponents are explained in detail in the following sections.

### Presentation Layer

UGameView class is at the base of the Presentation layer. It is responsible for displaying the system parameters and communicating user input to the Business Logic layer. UGameView communicates with a single GameTime instance that maintains all game parameters of the system. A variation of subpanels is developed to provide user interface including game menus, game view components and game object representations.

[INSERT DIAGRAM HERE]

### Domain Logic Layer

Business Logic layer is mainly characterized by GameTime and Game Controller classes. An instance of GameTime contains all runtime parameters of the system such as Map, Structures, and others. Presentation layer, namely UGameFrame, accesses GameTime to acquire system state and display it, but GameTime does initialize communication with the presentation layer. GameController listens for and handles user input communicated to it through the presentation layer and passes on the information to the GameTime instance that transmits required changes to individual system components. GameController is implemented as a Façade Controller as it is singular and represents the overall system.

[INSERT DIAGRAM HERE]

## Object Oriented Principles and Patterns

### Principles Used

### Pattern Used

# Dynamic Behaviour: Sequence Diagrams

# (System Logic…) Design

# User Interface Design

## Description of the User Interface

## Visual Presentation of Graphical User Interface

# References

**Daniel Sinnig PhD** Lecture Slides, ECSE-321. McGill University Winter 2015

**Martin Fowler** UML Distilled: A Brief Guide to the Standard Object Modelling Language.