Tower Shop

Architecture/Design Document

**Table of Contents**

[1 Introduction 3](#_Toc131876418)

[2 Design Goals 4](#_Toc131876419)

[3 System Behavior 4](#_Toc131876420)

[4 Logical View 5](#_Toc131876421)

[4.1 High-Level Design (Architecture) 5](#_Toc131876422)

[4.2 Mid-Level Design 6](#_Toc131876424)

[4.3 Detailed Class Design 8](#_Toc131876425)

[5 Process View 11](#_Toc131876426)

[6 Development View 12](#_Toc131876427)

[7 Physical View 12](#_Toc131876428)

[8 Use Case View 12](#_Toc131876429)

Change History

**Version:** A.2

**Modifier:** Nick

**Date:** 3/4/2020

**Description of Change: Finished file**

**Version:** A.1

**Modifier:** Nick

**Date:** 3/3/2020

**Description of Change: File was made**

# Introduction

**Architecture and Design**

The purpose of the architecture/design document is to explain the organization of the code. A well-written architecture document will make it easier for new programmers to become familiar with the code.

The architecture/design document should identify major system components and describe their static attributes and dynamic patterns of interaction.

Software architecture and designs are typically expressed with a mix of UML models (class and sequence diagrams being the two most common) and prose. Dataflow diagrams are also helpful for understanding the interaction between components and overall flow of data through the system.

**About this Template**

This template suggests one way of documenting a software system’s architecture/design. You aren’t required to include every section in this template nor all the content in the sections you do include. However, the document you do submit should pass the following checklist:

* Are design objectives clearly stated? For example, if performance is more important than reusability, this should be made clear at the start of the design specification.
* Does the architecture partition the implementation into clearly defined subsystems or modules with well-defined interfaces?
* Does the architecture express in a clear way the main patterns of communication between subsystems and modules?
* Does the architecture satisfy the requirements?
* Is the architecture traceable to requirements?
* Any models created should either be expressed with a well-known modeling language, or if a well-known modeling language isn't used, the syntax and semantics of the symbols that are used should be defined.

This document describes the architecture and design for the Tower Shop application being developed for Tilan. It is a UI menu that allows selection of towers that can then be placed in the world. It deals with purchasing and placing of towers.

The purpose of this document is to describe the architecture and design of the Tower Shop application in a way that addresses the interests and concerns of all major stakeholders. For this application the major stakeholders are:

* Developers – they want an architecture that will minimize complexity and development effort.
* Project Manager – the project manager is responsible for assigning tasks and coordinating development work. He or she wants an architecture that divides the system into components of roughly equal size and complexity that can be developed simultaneously with minimal dependencies. For this to happen, the modules need well-defined interfaces. Also, because most individuals specialize in a particular skill or technology, modules should be designed around specific expertise. For example, all UI logic might be encapsulated in one module. Another might have all game logic.
* Maintenance Programmers – they want assurance that the system will be easy to evolve and maintain on into the future.

# Design Goals

There is no absolute measure for distinguishing between good and bad design. The value of a design depends on stakeholder priorities. For example, depending on the circumstances, an efficient design might be better than a maintainable one, or vise versa. Therefore, before presenting a design it is good practice to state the design priorities. The design that is offered will be judged according to how well it satisfies the stated priorities.

The design priorities for the Tower Shop> system are:

* The design should minimize complexity and development effort.
* The design should allow for the ability to change on the fly. With items being changed around but it still working as it should.
* It should take away some of the logic from other systems. Works on its own with the other systems

# System Behavior

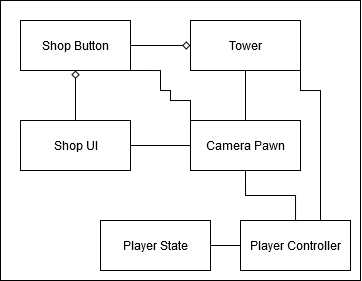
This system allows the user to interact with a UI based menu and the real world in game. The shop will populate itself with towers that are set in the blueprints. The player can click on one of the buttons that it creates and then click in the game world to spawn the selected tower. The shop will check how much money the plyer has before selecting a button to deactivate it. And upon click in the world it purchases the tower from the player.

# Logical View

The logical view describes the main functional components of the system. This includes modules, the static relationships between modules, and their dynamic patterns of interaction.

In this section the modules of the system are first expressed in terms of high level components (architecture) and progressively refined into more detailed components and eventually classes with specific attributes and operations.

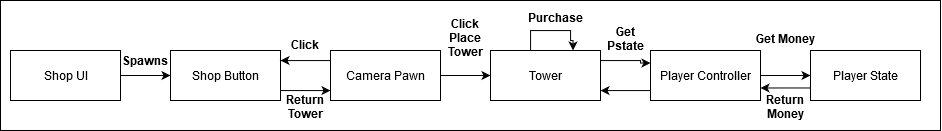
## High-Level Design (Architecture of the Shop system)



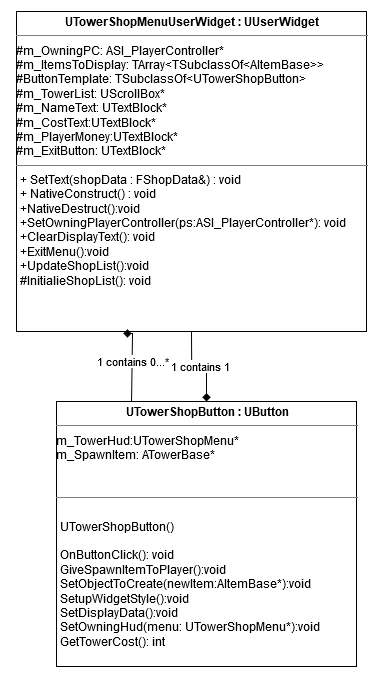
### System Architecture

* The **Shop UI** displays the information to the user. It also has an object that holds all of the shop buttons.
* The **shop button** is a custom button that has a reference to a tower. When the player clicks the button it sets the camera pawns active tower to the buttons tower.
* The **tower** in this system refers to the tower data. The tower data is what makes the tower the specific tower.
* The **camera pawn** is a pawn that provides a view of the play area from above and it also does the logic to place the selected tower.
* The **player controller** controls the pawn and has access to the **player state** that has the players money.

## Mid-Level Design of Module Tower Shop Menu



## Detailed Class Design of Module Tower Shop Menu



# Process View of Module Shop System

# 

# Physical View (Applies to Multiplayer)

The UI will be on the client’s computers. The client may calculate the tower it will spawn and the location and tell that information to the server

# Use Case View

The player clicks a button, the button displays the data and gives the CameraPawn a reference to the data for the tower to spawn, the camera pawn casts a ray to the world and then if it can place the tower the tower will take away the money and the world will spawn the tower at the location clicked.