University of Windsor

School of Computer Science
https://cs.uwindsor.ca

Master of Applied Computing

COMP-8117

Advanced Software Engineering Topics

Design Document

for

Silent Screamer

Version 4.0

Prepared by

Dhrumil Sanjaykumar Parikh	**110072217**
Prince Patel	**110072773**
Hetansh Sharad Shah	**110074680**
Yash Navneetbhai Bhalala	**110085755**
Ahmed Abdullah	**110068897**
Vrushit Nileshkumar Bharucha	**110073524**
Ahmed Mustafa	**110078623**
Dola Sai Siva Bhaskar Thota	**110088374**

March 24, 2022

Table of contents

1. Introduction

1.1 Purpose

The main goal behind writing this document is to give an overall idea about the functionality of the game Silent Screamer. It will also describe assumptions and dependencies, general constraints, goals and guidelines, and the development method which will be used to design the game. Apart from all of these, this document also contains architectural strategies which are being used to develop this game.

2. System Overview

This is a 3D desktop game that is based on a Horror theme, and it is a horror survival game. First, at the start of the game, the user will see the dark site where there is one mansion, and the user is outside of it.

When the user enters the mansion user does not have any weapon and the user can find weapons from different places. There will be an inventory where users can store the weapons and use them to kill the ghosts. Users can also find an apple on the site which can help user to increase their health. If the health of the player reaches to zero game will be over and the player will be thrown to the main screen.

While roaming around the mansion user will see multiple ghosts. Users can also change the modes such as turning on the flashlight and night vision. But when it is turned on there will be one indicator that has stats of battery percentage and when it reaches zero the flashlight will be turned off.

There will be specific points for killing the ghost when a user reaches a certain point one special weapon will be available in inventory and by using it user can kill the ghost in a single shot. And that weapon can only be used through voice command.

Apart from the mansion, there will be another site called the graveyard in which multiple ghosts will be present. To win the game user needs to kill every ghost present on both sites.

3. Design Considerations

3.1 Assumptions and Dependencies

- Mansion will be built (which will be the first map of the game).
- A graveyard will be built (which will be the second map of the game).
- A lobby screen will be there, which will be used as the main screen of the game.
- The player can walk, run (by pressing shift on the keyboard), and can jump.
- The player can use a flashlight and night vision for a limited time.
- The player can pick an apple, knife, and axe from the ground by pointing the cursor towards the weapon and pressing “e” on the keyboard.
- A secondary ghost will look around and run towards the player when the player will be in sight of a ghost.
- Secondary ghost hits will reduce the health of a player and after some hits, the player will die.
- The game will be lost when the player is dead and will return to the lobby screen.
- One primary ghost will be placed at each site.
- An inventory feature will be added to the game which should be displayed on the screen by pressing “I” on the keyboard.
- Weapons will be selected from inventory and will be shown in the first-person hand.
- The animation will be applied to weapons.
- When attacked by any weapon, the animation will be applied to a secondary ghost who will act as if he got hit by something.
- The health of secondary ghosts will be reduced while getting hit by a weapon.
- Spam of hitting will be controlled in first-person by adding hitting stamina in the script.
- The point system will be added to the game which will reflect on the killing of the ghosts.
- A specific number of points will enable a special weapon.
- A special weapon will appear in inventory when the player reaches a specific number of points.
- The special weapon will work through the voice command of “Shoot”.
- Ghost will die with only a single hit by the special weapon.
- A secondary ghost will fall and die after getting killed by the first person.
- Secondary ghost will disappear after 25 seconds when dead.
- When all ghosts are dead, the player will win the game.
- Working Deliverable model of the game for each iteration will be ready to be tested by the customer after each milestone.
- Project cost will remain the same as per the initial estimation.
- The scope of the project will remain unchanged throughout the development.
- The project will follow an agile methodology throughout its execution.
- A dependency exists with the unity engine.
- Unity is chosen as an engine because of the wide range of community support.

- The C-sharp language will be used to program/write code in the unity engine, as a unity only supports c-sharp language ***natively*** and all unity engine libraries are built using c-sharp.
- Customer will test and give feedback.

3.2 General Constraints

- System (Laptop/PC) constraint in development.
- The project file should be uploaded on GitHub using git, to make it possible for other developers to work on the same project.
- Tracking and assigning tasks on Jira.
- The game should run in Windows 7 and higher.
- Code should be maintained by documenting the code.

3.3 Goals and Guidelines

Goals:

Horror survival game where you explore a mansion and graveyard to try to find several types of weapons and ghosts.

- Survival and Investigative horror
- Storyline
- First-person
- Environment design
- Weapons Development
- Night vision tools development

Minimal Interface The game is built on 4 design goals:

1. Discovery - It may be argued that horror is less about dread and more about discovery, such as seeking ghosts in a solitary location with terror as a motive - the idea is to communicate the experience of discovery throughout the game.
2. Fear - The player should always be on edge, never feeling safe, and knowing that someone is watching them, creating a constant and unending sense of tension - as well as more traditional fear when desperately fighting the enemies the mansion throws at them while being given just enough resources to feel fearful.
3. Triumph - When they win a battle by a hair's breadth or reach a major milestone in their quest.
4. Minimalism - Minimal user interface elements to aid immersion and force the player to keep a careful check on their resources.

Guidelines:

- Characters cannot jump out of windows
- The game starts with the first player being just outside the mansion with no weapon in hand.
- Characters can only use only one weapon at a time and cannot use multiple weapons at a time.
- The player can pick up the weapon with the specified control.
- Characters cannot run through walls or doors that are closed
- The Player's view is purely over the shoulder of their character – no 360 degrees
- The player starts with no weapons and collects weapons on the way
- The player can enter the mansion or the graveyard site.

3.4 Game Development Description:

- Environment:
    - The location of the game is the Mansion, which is the central building of the map.
    - There is one greenhouse and one other house in the center of the map.
    - Trees and fences are there surrounding the Mansion area.
    - Blockers are used to create a closed world.
    - There is one floor, and each floor has multiple rooms in the Mansion.
    - The graveyard and Mansion are on the same map.
    - The graveyard has multiple graves, and it has an opening from which players can enter it.
- Characters:
    - Player Characters:
        - Currently, there is only one character in the game who is by default selected.
    - Non-player Characters:
        - Ghosts
            - Secondary Ghost – Easier to kill
            - Primary Ghost – Harder to kill
- Collectables:
    - Weapons:
      The weapons are used to kill the ghosts.
        - Knife
        - Axe
        - Sword
    - Apple (to increase health)
      An apple is an object which will be used to increase health

3.5 GamePlay I/O Controls:

- Keyboard
- Mouse
- To move around the environment:
    - Arrow keys or W, A, S, and D keys
- Special keys:
    - Space = jump
    - E = Pick up objects
    - F = To use flashlight
    - N = To use Night vision
    - Left-click = To attack the ghost using the weapon

3.6 Visual and Audio Features:

- Visual Features:
    - Bottom Centre – Health bar
    - Bottom Left – Battery life
- Audio Features:
    - Footsteps
    - Jumping and Landing
    - Environment noise

3.7 System Parameters & Requirements:

- Keyboard
- Mouse
- Monitor
- 2 GB RAM
- Dual-core processor
- Speakers
- Hard drive
- Graphics memory
- Windows XP or Windows Vista or Windows 7
- DirectX

4. Architectural Strategies

Architectural strategies used in this project are based on the sprint and iterations used in the planning and design of the game. The main strategy is placing the game as the first-person user game which would follow objects created and made around the user. This is based upon an Event-driven architecture in gameplay where the user would interact with incoming objects and items which indirectly or directly affect the user's behavior in the game. As a planned horror game, our system would provide much communication with the environment.

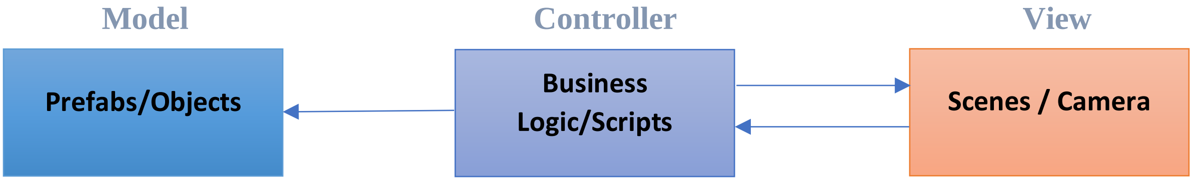
Strategies used in the project:

- Planning and Designing tasks in Sprint and Iterations
- Actions in first-person user game
- Interacting objects for the user in the game
- Merging the contents with Maps and environment
- Bundling the game for the run at the end of the iteration cycle
- Testing the character and object interactions
- Environment and Maps synchronization
- Ghosts on the site
- User weapons and side things
- Sound effects and sensing on the Screen of the user
- Documentation of all the techniques and methods used in the project

5. High-Level Design

5.1 Architecture

The architecture we have used in our project is MVC (Model, View, Controller).

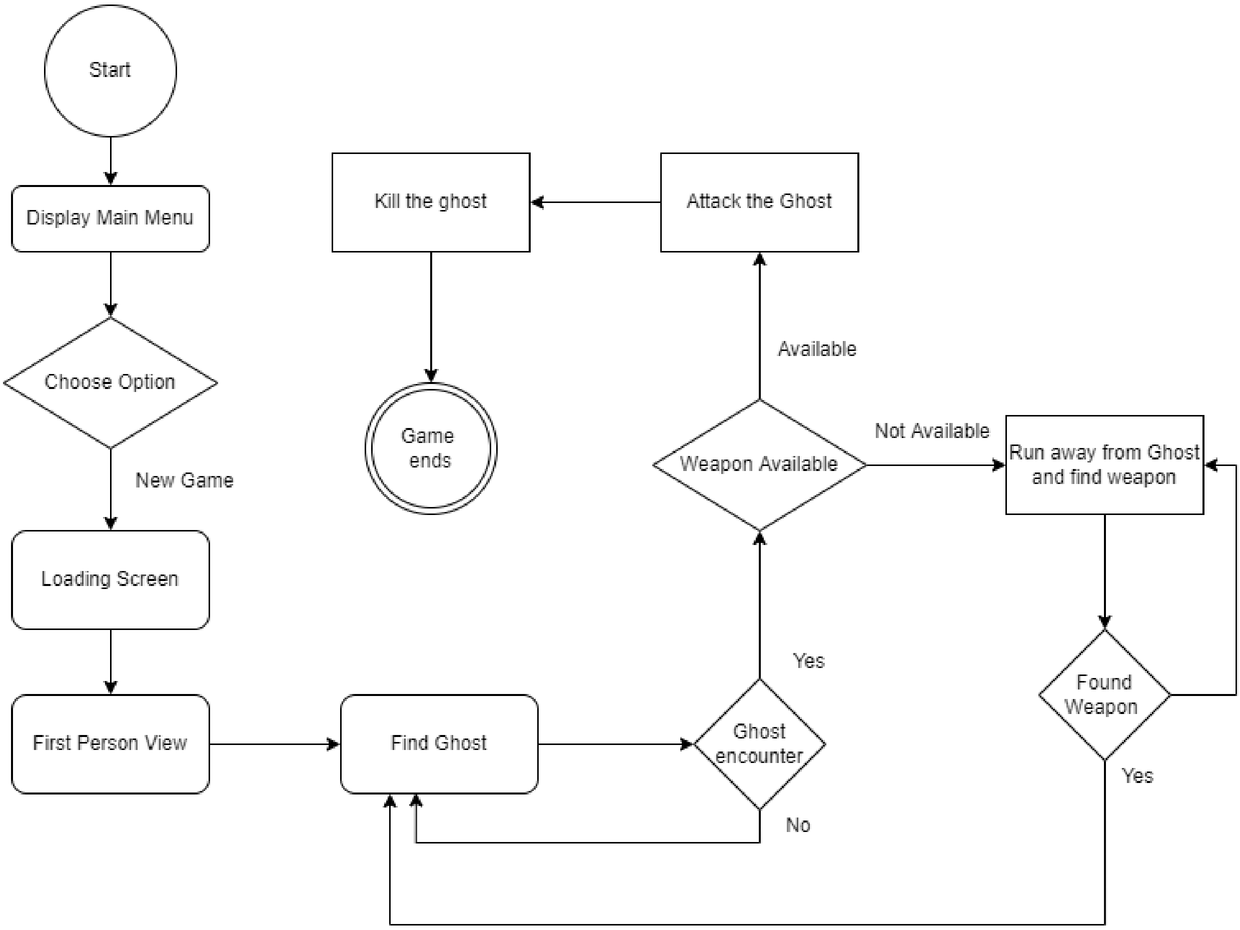


The Unity Monobehaviours are in the View layer of MVC which is supposed to shield the architecture from the hard elements. View layer creates instances of uses (SerializeField) and prefabs which are used to drag and drop components. There is no logic in the View layer, just visualization of data is there.

The Controller layer has the business logic and controls the animation, sound, movement, and all the logic. It does not depend on view nor defines the way to store data (which is the work of the Model layer). The Controller layer communicates with the Model layer as well as the View layer.

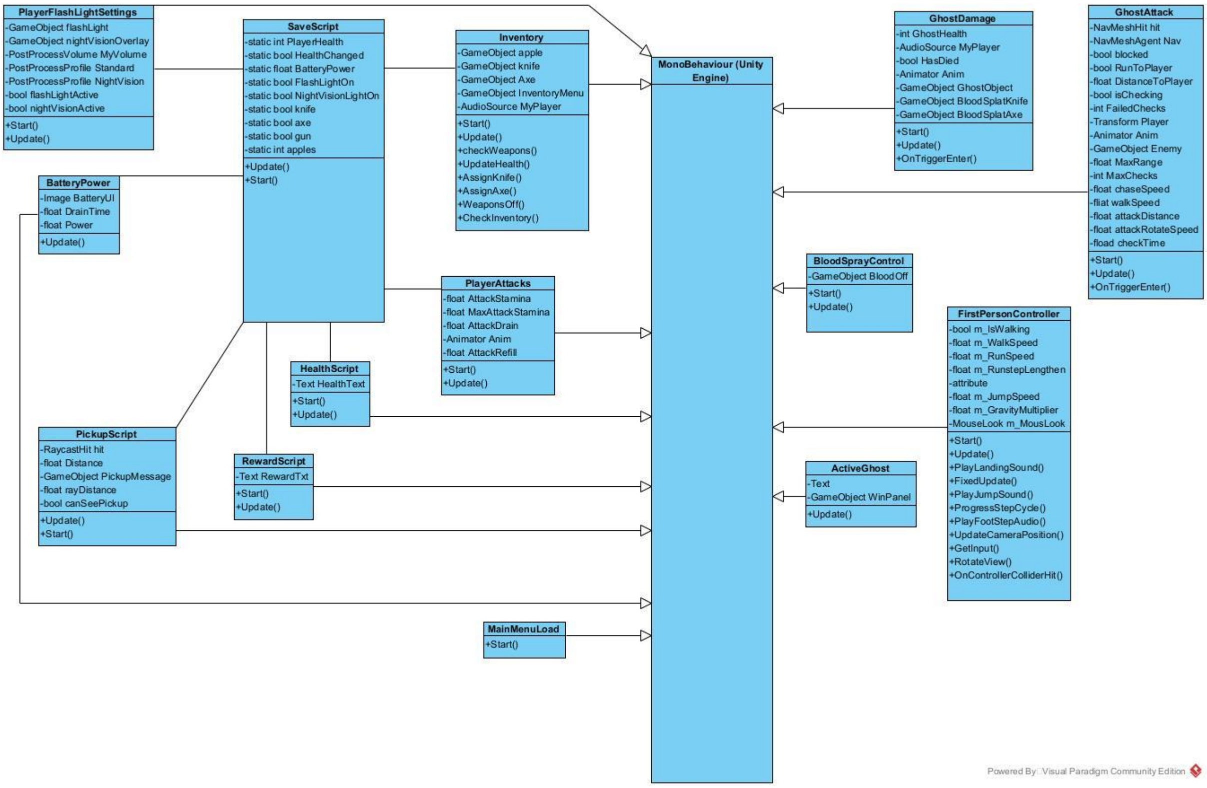
The Model layer is a layer where all data operations are performed. Retrieving characters, items and weapons is one example of data operations.

5.2 High Level Design

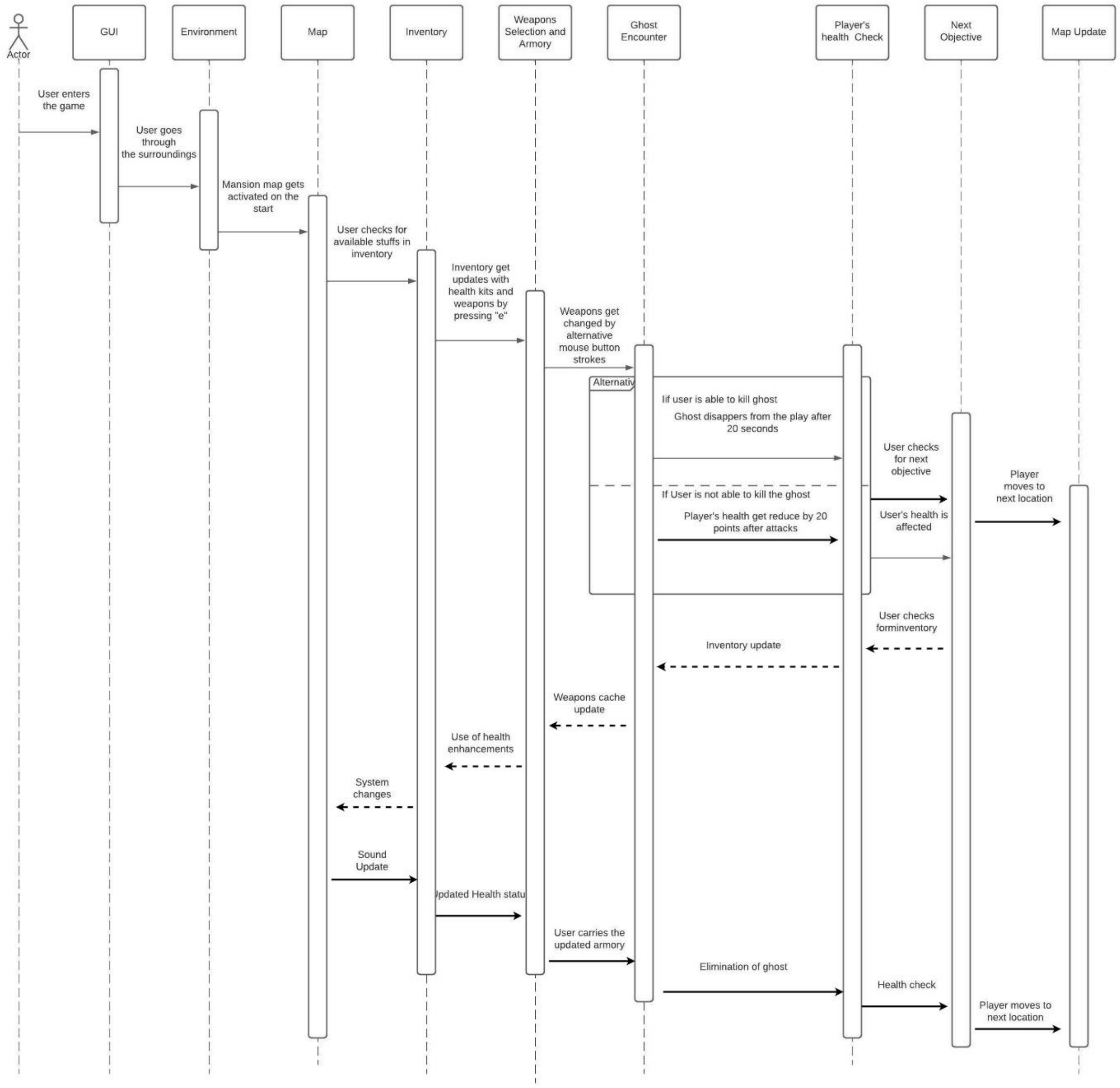


6. Low-Level Design

6.1 Class Diagram



6.2 Sequence Diagram


Actor
GUI
Environment
Map
Inventory
Weapons Selection and Armory
Ghost Encounter
Player's health Check
Next Objective
Map Update
User enters the game
User goes through the surroundings
Mansion map gets activated on the start
User checks for available stuffs in inventory
Inventory get updates with health kits and weapons by pressing "e"
Weapons get changed by alternative mouse button strokes
Alternativ
Iif user is able to kill ghost
Ghost disappers from the play after 20 seconds
User checks for next objective
Player moves to next location
If User is not able to kill the ghost
Player's health get reduce by 20 points after attacks
User's health is affected
User checks forminventory
Inventory update
Weapons cache update
Use of health enhancements
System changes
Sound Update
Jpdated Health statu
User carries the updated armory
Elimination of ghost
Health check
Player moves to next location