МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра програмних систем і технологій

ЗВІТ

про проходження переддипломної практики в
умовах кафедри програмних систем і технологій

Індивідуальне завдання: Software Architecture Document (SAD)

Виконав: ст. гр. ІПЗ-43
Козубенко Сергій Сергійович

Керівник практики:
к.ф.-м.н., доцент Карнаух Тетяна Олександрівна

Київ – 2023

Software Architecture Document

content owner: Kozubenko Sergiy

Document Number: 1 Release version: 1.0 Release Date: 04-30-2023

Contents

Introduction

This SAD provides a detailed overview of the software architecture for the development of a 2D fighting game with elements of martial arts using C# in the Unity platform. This document will provide an overview of the game's functionaland non-functional requirements, software architecture, design patterns, and otherimportant aspects of the game development process.

Object

This SAD serves as the primary reference for the software architecture of the 2D fighting game. It will be updated as necessary throughout the development process to reflect any changes in the system's architecture. The document will be controlled using version control software

Purpose and Scope

The scope of this document includes the design and architecture of the 2D fighting game with martial arts elements, including the game mechanics, graphics, user interface, and code architecture.

This document does not include the implementation details of the game, such as specific code or asset creation.

The purpose of this SAD is to define the software architecture for the 2D fighting game, providing a detailed description of the system's major components, their interactions, and their interfaces. This document is intended for use by software developers who will be implementing the system

Requirements

The game will feature 2D graphics with pixel-style characters and environments.

The user interface will include menus, character select screen, health bars, and other gameplay elements.

The player will be able to select a character, each with their own unique moves and abilities.

The gameplay will feature basic attacks, special moves, combos, and other fighting game mechanics.

Viewpoint Definitions

Logical Viewpoint: This viewpoint describes the system's major components and their interactions from a logical perspective, including the overall structure of the software and the flow of data between different components.

Physical Viewpoint: This viewpoint describes the system's major components and their interactions from a physical perspective, including the deployment of the software and the distribution of its components across different hardware systems.

Development Viewpoint: This viewpoint describes the system's major components and their interactions from a development perspective, including the organization of code modules and the development processes used to build the software.

Architectural Approach

Architecture Background

This section provides a background on the architecture of the 2D fighting game.

Problem Background: The 2D fighting game is designed to provide players with a challenging and engaging fighting experience, with a variety of characters, moves, and environments to choose from.

System Overview: The 2D fighting game consists of several major components, including the game engine, graphics renderer, physics engine, and sound engine.

Goals and Context: The goal of the 2D fighting game is to provide an immersive and engaging fighting experience for players, with responsive controls, smooth animations, and realistic physics.

Significant Driving Requirements: The 2D fighting game must be able to handle multiple players, with responsive controls and smooth animations, while maintaining a high frame rate.

Solution Background: The 2D fighting game is built on the Unity platform, using C# as the programming language.

Software Architecture

The game will be developed in Unity using the C# programming language.

There are several architectural approaches that can be taken for the development of a 2D fighting game with elements of martial arts in the Unity game engine using C#. Some of the common approaches include:

The code architecture will follow the Model-View-Controller (MVC) design pattern. Model-View-Controller (MVC) architecture: This approach involves separating the game's data, user interface, and control logic into three different components. This separation helps to promote a clear separation of concerns and makes it easier to maintain the codebase over time.

Component-based architecture: This approach involves creating reusable components that can be attached to game objects. Each component is responsible for a specific task, such as handling player movement, managing AI behavior, or handling collision detection.

Entity-Component-System (ECS) architecture: This approach is similar to the component-based architecture, but instead of attaching components to game objects, the game objects themselves are merely collections of components. The ECS architecture is designed to be highly scalable and performant, making it a popular choice for large-scale game development.

After considering the requirements of the game and the pros and cons of each architectural approach, it was decided that a component-based architecture would be most suitable for this project. This approach provides the necessary flexibility to create a variety of different game objects with unique behaviors and enables a high level of code reuse.

The game will use a state machine to manage the game's states and transitions.

Design Patterns

The game will use the Observer pattern to handle events and communication between different components.

The game will use the Singleton pattern to ensure that only one instance of certain components is active at a time.

Non-Functional Requirements

The game will have a maximum file size of 500 MB.

The game will have a loading time of no more than 10 seconds.

The game will have a minimum frame rate of 60 FPS.

The game will be compatible with Windows operating systems.

Conclusion

This Software Architecture Document has provided an overview of the architecture and design of the 2D fighting game with martial arts elements.

The document has outlined the software architecture, design patterns, and nonfunctional requirements of the game, providing a roadmap for the game development process.

Analysis Results

The analysis phase of the SAD helped to identify the key stakeholders and their concerns, the system's goals and context, significant driving requirements, and the architectural approach to be taken. The following are the key results of the analysis phase:

Stakeholders: The key stakeholders in this project are the game developers, game testers, and end-users who will be playing the game.

System goals and context: The system goals are to create an engaging 2D fighting game with elements of martial arts that is enjoyable to play and meets the

expectations of the target audience. The context of the system is the Unity game engine, which will be used to develop the game.

Driving requirements: The driving requirements for this project include creating responsive and fluid gameplay, designing engaging levels with varied challenges, and implementing a range of different martial arts moves and techniques.

Architectural approach: A component-based architecture will be used for this project to enable the creation of reusable components and provide the necessary flexibility to create a variety of different game objects with unique behaviors.

Requirements Coverage

The requirements for the game were defined in the project's initial requirements document and have been analyzed and refined during the SAD process. The component-based architecture that has been chosen is capable of meeting all of the game's requirements, including:

Creating responsive and fluid gameplay

Designing engaging levels with varied challenges

Implementing a range of different martial arts moves and techniques

Creating a variety of different game objects with unique behaviors

Referenced Materials Directory

The following materials were referenced during the creation of this SAD:

Unity documentation: The official documentation for the Unity game engine was used to gain a better understanding of how to implement certain features and components.

C# documentation: The official documentation for the C# programming language was used to gain a better understanding of certain language features and syntax.

Martial arts resources: Various online resources were used to research different martial arts moves and techniques, which will be incorporated into the game.

Glossary

2D: A two-dimensional graphics environment, as opposed to 3D graphics.

C#: A programming language developed by Microsoft, commonly used in Unity game development.

Combo: A sequence of moves performed in quick succession, often resulting in increased damage to the opponent.

MVC: The Model-View-Controller design pattern, a widely-used architectural pattern in software development.

State machine: A design pattern used to manage the states and transitions of a program or system. Acronyms

FPS: Frames Per Second

MVC: Model-View-Controller

SAD: Software Architecture Document References

"Model-View-Controller (MVC)," Microsoft Docs, accessed April 30, 2023, https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions1/overview/understanding-models-views-and-controllers-cs.

Procedural Content

Generation for Unity Game Development

Procedural Content Generation is a process by which game content is developed using computer algorithms, rather than through the manual efforts of game developers. This book teaches readers how to develop algorithms for procedural generation that they can use in their own games. These concepts are put into practice using C# and Unity is used as the game development engine.

Unity Documentation

https://docs.unity3d.com/2023.2/Documentation/Manual/index.html Game Programming

Patterns Paperback – November 2, 2014

Useful patterns for game programming.