<Name-of-Software-Application>

CS 230 Project Software Design Template

Version 1.0

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	03/20/22	Jovi Billiot	Game software that allows multiple teams and players but only one game instance.

Instructions

Fill in all bracketed information on page one (the cover page), in the Document Revision History table, and below each header. Under each header, remove the bracketed prompt and write your own paragraph response covering the indicated information.

Executive Summary

The software is designed to only run one game at a time, however multiple teams can be added along with multiple players for each team.

Design Constraints

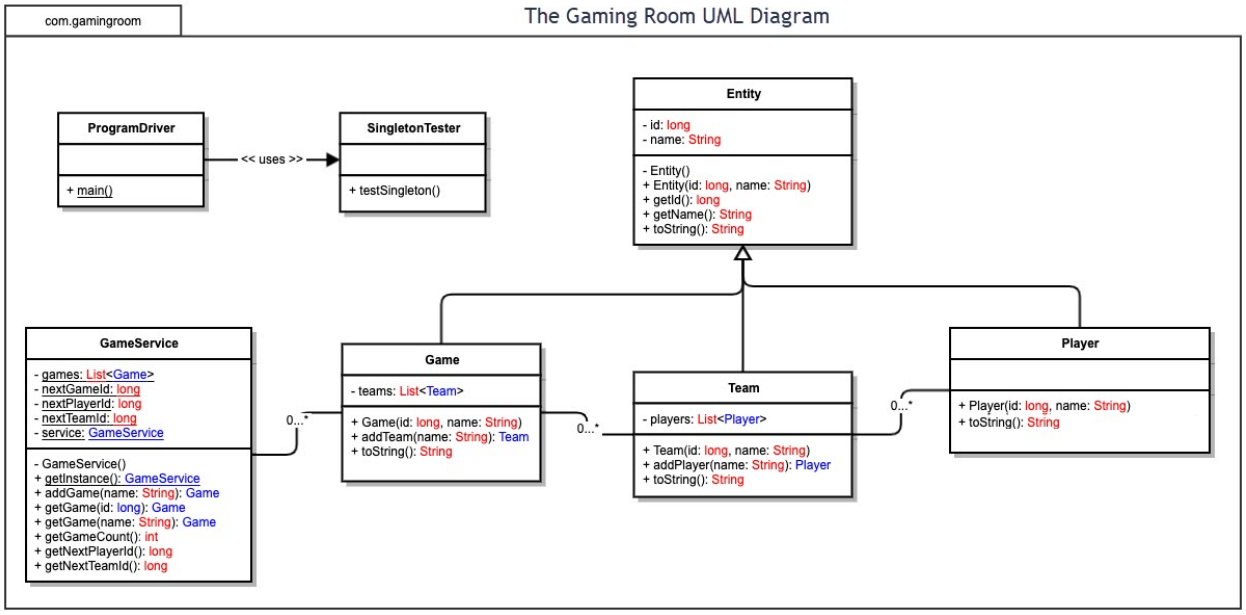
<Identify the design constraints for developing the game application in a web-based distributed environment and explain the implications of the design constraints on application development.
The design constraint of developing the game application in a web-based distributed environment is to ensure the application can operate on a consistent level across all operating systems. The application is written in Java which is a programming language that is recognized throughout all operating systems.

System Architecture View

Please note: There is nothing required here for these projects, but this section serves as a reminder that describing the system and subsystem architecture present in the application, including physical components or tiers, may be required for other projects. A logical topology of the communication and storage aspects is also necessary to understand the overall architecture and should be provided.

Domain Model

The Entity class holds all the methods and variables that the Game, Team, and Player classes share, so redundant code is eliminated by giving the ability of using those methods and variables to the other classes. The game service class generates a new game and ensures that only one game can be implemented at a time. The game class adds teams to the game, and the team class adds players to the team. The Program Driver class's main method is where all the code come together, and it call upon the Singleton Tester class to ensure only one game can be ran at a time.



Evaluation

Using your experience to evaluate the characteristics, advantages, and weaknesses of each operating platform (Linux, Mac, and Windows) as well as mobile devices, consider the requirements outlined below and articulate your findings for each. As you complete the table, keep in mind your client's requirements and look at the situation holistically, as it all has to work together.

In each cell, remove the bracketed prompt and write your own paragraph response covering the indicated information.

Development Requirements	Mac	Linux	Windows	Mobile Devices
Server Side	Security is an advantage to Mac. Apple does a good job of making sure your privacy is safe from outsiders. I haven't found any major disadvantages to the server side of Mac.	I do not have any experience with Linux, but I do know it is open source. The open source aspect of Liunx makes it super secure because it is constantly being worked on and updated.	<Evaluate Windows for its characteristics, advantages, and weaknesses for hosting a web-based software application.> A disadvantage to Windows compared to the other operating systems is, it's more susceptible to viruses. It is still a very popular operating system that supports a very large variety of web-based applications.	Mobile devices are constantly getting better in their ability to be just a useful as the other operating systems. They are smaller, so they can fit as much information on the screen, and its harder to navigate through large websites. Some websites don't have a mobile version, so sometimes its easier to just use a different operating system. For any web application that does have a mobile version, it is almost guaranteed that a mobile device will be the most convenient option.

Client Side	Macs are expensive products, however their efficient once you get use to them. There is many features of a Mac, but I believe in order to realize the full potential of a Mac, you must be fairly computer savvy. For someone who doesn't do much on a computer, a Mac can be considered a waste of money.	<Determine the software development considerations (cost, time, expertise) that are necessary for supporting multiple types of clients as they pertain to Linux.> I've never used Linux, so I'm not sure when the client side is like.	Windows is the operating system used on most laptops; therefore, you can find more affordable windows operated computers. Window is great for people who spend most of their time on their computer, and people who pend very little time on their computer. I find it easier to navigate around a windows operating system than the others.	Sone mobile devices can cost much less than a laptop, however they can get pricy. They are very user friendly and customizable. Out of all the other operating systems, they are the most convenient to use.

Development Tools	Swift is the language that is mostly used for applications for Macs. Max also has its own IDE called XCode that supports multiple programming languages. Most IDEs will work for Mac as well.	I don't have any experience in this matter.	Windows supports most operating systems and programming languages. Unlike Mac, visual studio is a viable option for Windows users when writing C++ programs.	<Identify the relevant programming languages and tools (IDEs and other tools) that are used to build this type of software for deploying on Mobile Devices.> Depending on the operating system of the mobile device, will determine the language and tools used to deploy applications on them. Java is mostly used for android applications but can be used for Apple. Swift is mostly used for Apple devices. Kotlin has become a big player in building Android applications as well.

Recommendations

Analyze the characteristics of and techniques specific to various systems architectures and make a recommendation to The Gaming Room. Specifically, address the following:

1. **Operating Platform**:
   I recommend launching the game as an IOS Application. iPhones are extremely popular devices that will give the game exposure to soon expand to other computing systems. IOS is a more secure platform than others, which makes it a great place to operate.

2. **Operating Systems Architectures**:
   IOS runs a layered architecture. The different layers work together while serving their own purpose, which creates efficiency. There is layers for interface, graphics, data transfers, audio, and communication.

3. **Storage Management**:
   Part of the core layer of the IOS architecture, there is a framework that transfers data from the application to the cloud and vice versa.

4. **Memory Management**:
   The iOS platform has a built-in system that controls memory by freeing data as needed to prevent memory leaks and applications from crashing.

5. **Distributed Systems and Networks**:
   The iOS platform extends to Apple products, so users with MacBooks and iPhone could enjoy Draw it or Lose it, however the game is written in Java, so it could expand to all platforms. The game could reach various cloud systems that may reach multiple platforms and users.

6. **Security**:
   The game can be secure throughout all platforms because users are only allowed minimum access to the application. Someone playing the game may only access their profile and the gameplay, whereas admins have access to their private features but not user profiles. These restrictions are built into the code of the game application.