

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 02/19/2023 | Keri Ludemann | Allow multiple teams to play during one game, loops to keep team names unique, only one game should exist at one time in memory. |

**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](#_sbfa50wo7nsh)

The client wants one game to exist in memory at a time and must only allow unique ids. There can be multiple teams playing per game. The game needs to show stock images to players for them to guess what is being drawn.

## Requirements

*<* Please note: While this section is not being assessed, it will support your outline of the design constraints below. *In your summary, identify each of the client’s business and technical requirements in a clear and concise manner.>*

## [Design Constraints](#_2et92p0)

Only one game in memory at a time, multiple teams of multiple people with unique names and ids, and needs to be available on multiple platforms.

## [System Architecture View](#_ilbxbyevv6b6)

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](#_8h2ehzxfam4o)

The ProgramDriver class has the main function which is used to execute the program and the class also uses the SingletonTester class. The Player class references the Team class which references the Game class and that references the GameService class. The Game, Team and Player class inherit from the super class Entity.

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## [Evaluation](#_2o15spng8stw)

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** | Mac is flexible when it comes to configuring the server and the second most used operating system. | Linux is the least used out of the operating systems but offers the same flexibility as Mac at a lower cost. | Windows is the most commonly used operating system and has a wider variety of platforms powered by it. | Mobile devices are owned by many individuals but Apple and Android use different operating systems. |
| **Client Side** | Mac is not open source so cost could be similar to windows. It will require expertise and time. | Linux is open source so there are no license fees. With Linux being a bit difficult compared to Windows and Mac it would require the most time and expertise. | Windows is not open source so there will be a cost associated with using it. This OS would require the least amount of time and expertise. | Since the major mobile devices use different systems to operate it will take more time when dealing with them. Android is open while Apple is closed meaning there will be a cost associated with using it. |
| **Development Tools** | The main coding language for Mac seems to be swift but others are used as well such as C, Python and Javas. There are multiple IDEs available for use. | Apache Netbeans can be used for Linux along with IntelliJ IDEA. Both use Java. | Visual studio seems to be popular for Windows and it uses C++ along with a few other languages. | Apple products will use the same language and IDE as Mac but Android will use IntelliJ IDEA like Linux. |

## 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**: Based on information that has been provided, I would say that Windows is the best route to go since they have the widest range of devices even with the associated cost. Windows is also the most used platform with an abundance of IDE’s.
2. **Operating Systems Architectures**: Windows utilizes a hybrid kernel architectural system which combines aspects of both microkernel and monolithic kernel. This allows the system to have the design and speed of a monolithic kernel while also possessing the stability of a microkernel . Reference: https://www.geeksforgeeks.org/kernel-in-operating-system/
3. **Storage Management**: Cloud based storage is recommended due to the ease of access. With Cloud based storage you can access stored files on multiple devices, as long as you use the same profile , and you can easily gain more storage when necessary. The cost of storage is usually associated with a monthly subscription.
4. **Memory Management**: Windows utilizes virtual memory. This system keeps processes from interfering with each other by using virtual address space processes, uses file mapping to associate file contents with virtual address space of process, and comes with its own security. Reference: https://learn.microsoft.com/en-us/windows/win32/memory/about-memory-management
5. **Distributed Systems and Networks**: if the client were to use a network that supports a distributed system players would then be able to play with others across the world. The aspects of a distributed system, such as fault tolerance, scalability, etc, tend to keep the system from having major problems from happening and/or deeply effecting system. Reference: https://www.geeksforgeeks.org/what-is-a-distributed-system/
6. **Security**: Windows has built in security although, this adds little protection for user information. The cloud storage system also comes with its own protection. This mainly only protects the system side however and the end user(s) will need to have their own type of protection set up. This could be protection to keep the user’s account from being exploited.