

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 | 08/03/2024 | Quinton Hesse | Moved from Template to functional Document |

**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 Gaming Room has contacted Creative Technology solution to expand their gaming app DRAW IT or LOSE IT to Linux, Mac, Windows and Mobile. DRAW IT or LOSE IT is based on a 1980’s television game called Win, Lose or Draw. In this game teams compete to guess on what is being drawn. The game application renders images from a library of stock images that render in 30 second intervals. If the team doesn’t guess by the end of the 30 second render, the opposing team gets 15 seconds to offer one guess.

## Requirements

Compatibility: the server that is hosting the game must be compatible with clients from other operating systems. This will require a unified XML code.

Functional/Security: The Server must be able to authenticate users from all different platforms ensuring all users have paid for the game.

Usability: the rendering of the image must be synced across all platforms to ensure fairness of play.

## [Design Constraints](#_2et92p0)

Technical: the gaming application must be able to run with the limited resources of the mobile platforms and older personal computer systems.

Talent: since the gaming room is requesting help in developing for these systems their developers may not be familiar with these systems

Technical: The Game absolutely must be synchronized despite internet latency, connectivity and bandwidth to ensure fairness of play.

Financial: cost of renting a server, with ability to scale up or down depending on popularity of the game and demands on server.

Financial: cost of one or more development teams and the tools they need to develop the game.

## [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 program driver uses the singleton tester to test the functionality of the singleton design in the main method. Game service is 0 to many with the Game class. Game is 0 to many with Team class. Team is 0 to many with Player class. Game, Team and Player inherit from Entity. Entity is the parent class, with Game, Team and Player being child classes.

<Describe the UML class diagram provided below. Explain how the classes relate to each other. Identify any object-oriented programming principles that are demonstrated in the diagram and how they are used to fulfill the software requirements efficiently.>

**"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** | MacOS include Apache web server software and is great for print, media, mail, files, and handling other IOS devices, less feature complete than Linux or windows. as of 2022 mac discontinued macOS server, however users can continue to use the older versions. There are companies that rent out MAC OS server deployments. MacOS servers can be hosted on hardware or in the cloud, the cost varies depending on whether newer more powerful apple hardware is used and how many systems are used. | Linux is open source and has a wide range of different programs and just about any hardware can have Linux installed on it. Linux is the most flexible option with the most variety of solutions. Cloud based solutions are plentiful for Linux, a major player is amazon web services, with their gamelift program pricing goes from $4,628 per month from less demanding games with a peak player count of 10,000 player  <https://aws.amazon.com/gamelift/pricing/>  Linux Hardware would be a bigger upfront cost but may be more economical in the long run, after the initial cost only the electric bill is required, until a performance upgrade is needed | Windows is another solid deployment method with many services in place. Microsoft itself has a cloud-based server hosting for games called Microsoft Azure Playfab, which has a pay as you go meter system that scales up as the player count increases, this may be a good option if the popularity of the initial launch of the game may be hard the measure.  <https://playfab.com/pricing/>  Hardware side Microsoft has a very wide range of systems that can run it, and many different software solutions. | This option could work if each game was hosted locally without a large server infrastructure on the backend. Obviously, there are not many hardware or cloud solutions for backend server infrastructure that uses mobile devices to host games at scale. |
| **Client Side** | Developing for Mac OS would require the most amount of cost time and expertise for a medium customer base. since it will be browser based instead of hardware based IOS safari is built into macs and is often not compatible with even most HTML based games played through browser. the game is based in java there may be specific browsers that may not work. | Linux is open source and may have the most options for browsers compatible with a java game. Such as Firefox ESR. Linux would still be a very small customer base, as it doesn’t have a large share of the market despite it being free, available and open source. Linux comes with no built-in browser, so third party browsers would be the focus here such as Firefox or google chrome. | Windows would be the largest share of the market with the classic internet explorer being compatible with java browser games. The newer Microsoft edge comes with newer Microsoft models and is generally compatible with most browser games. Despite edge coming with windows, google chrome still holds more than fifty percent of browser use with safari coming in second, and edge in third. Instead of developing for a specific operating system client, the real goal should be to develop the browser game for google chrome. | Natively java cant run on IOS or android without emulators developing for mobile would be two fold IOS and Android. This would mean two different development teams with expertise in the two different mobile platforms. This may mean a higher cost but also the mobile market is one of the largest. |
| **Development Tools** | Since we would not be deploying on mac but rather its built-in browser which is safari special consideration must be taken to get safari to run HTML5 without compatibility errors. This development could probably be done in Eclipse Maven without licensing fees. | Since Linux users are the most likely to be using google chrome so in developing for chrome typically browser games can be developed with HTML, CSS, and JS. Chrome has the most users and should be high priority. This could be done on WebStorm for Linux with a $159 per user per year license. | Since edge is built into windows developing for edge would be a lot of work for the least number of users, special considerations would have to be made to ensure compatibility of the HTML or Java game on edge. This could be done with Microsoft’s VS code for $45 per use per month. | Xcode could be used to develop the game for IOS mobile market, as Xcode comes with many tools for developing for the iPhone. Xcode is free however the registration cost to publish your app on the Appstore is $99 a year. Development on android could be done with android studio or eclipse for free. |

## 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 would recommend windows to be the main platform for Draw it or Lose it, I feel that it has the most support given Windows dominance of the market and most programmers will be familiar with windows. Windows also has Visual studio which is a powerful IDE for developing games. Also Microsoft provides Azure Playfab if The Gaming Room didn’t want to host the game on their own hardware. Windows also has support for a wide range of hardware. Since Windows is the most used software it should experience the least amount of compatibility issues. Microsoft also has the most support, and technical documentation available for hosting a game server. Windows is tried and true and presents the most opportunities of all the platforms.

<Recommend an appropriate operating platform that will allow The Gaming Room to expand Draw It or Lose It to other computing environments.>

1. **Operating Systems Architectures**: Windows is closed source, hybrid kernel, modern windows versions are based on Windows NT and support 64-bit computing. Windows takes a layered design pattern, this makes the kernel more easily protected, due to the separation from the user. Windows has many built in features for hosting game servers like .NET Framework, and Active directory integration which enables The Gaming Room to manage multiple game servers. Windows also provides Performance and resource monitoring to identify bottlenecks and keep track of resources. Windows also provides tools for debugging.

<Describe the details of the chosen operating platform architectures.>

1. **Storage Management**: newer Windows versions use Indexed storage for its files called New Technology File System (NTFS), older versions may use a file allocation table. Indexed allocation puts all the pointers in a index block to allow for efficient and direct access. Windows also features Dynamic volumes with RAID features which could help for server hardware for hosting storage that’s complex and spans multiple disks.

<Identify an appropriate storage management system to be used with the recommended operating platform.>

1. **Memory Management**: Windows has virtual memory management such as paging that switches inactive pages to disk in favor of active processes. This will help optimize the amount of memory available to the game and will also be able to operate using a page file even if memory is full which will prevent out of memory errors. Windows also has virtual address space, which will help keep other programs from interfering with the game’s memory. Windows also features SuperFetch and prefetch which will preload frequently accessed game data into RAM. Windows also provides efficient heap and stack management ensuring avoidance of fragmentation and avoidance of stack overflow.

<Explain how the recommended operating platform uses memory management techniques for the Draw It or Lose It software.>

1. **Distributed Systems and Networks**: Draw it or Lose it would have to have features to account for latency, or the time from when a user does something to when that packet gets to the server. The server would have to manage the instance to account for latency so all players would have a reasonably fair experience. The software would also have to account for loss in connection of a player. Windows server provides features such as Quality of service which helps balance network loads. Windows also provides clustered servers and load distribution across multiple networked servers to distribute loads and minimize downtime.

<Knowing that the client would like Draw It or Lose It to communicate between various platforms, explain how this may be accomplished with distributed software and the network that connects the devices. Consider the dependencies between the components within the distributed systems and networks (connectivity, outages, and so on).>

1. **Security**: Kernel is in a different layer than user which helps protect the essential programs. Memory is protecting by securing processes from using other processes memory. Windows controls access to system resources and has file permissions and built in firewalls and antivirus. The servers hosting the game should consider masking IP addresses from other player to prevent players from obtaining each other’s IP addresses which could then be exploited. Windows provides data encryption to safeguard against unauthorized access.

<Security is a must-have for the client. Explain how to protect user information on and between various platforms. Consider the user protection and security capabilities of the recommended operating platform.>