

<Name-of-Software-Application>

# **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 |
| --- | --- | --- | --- |
| 1.2 | <02/6/23> | <Michael Russell> | <Completion of 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)

Design Problems as indicated by the client:

* Replicate 1980s TV show Win, loose, draw
  + Use stored images within a library so that each team can solve a puzzle
* One or more teams can be involved
  + Each team can have multiple players
* Game and team names need to be unique
  + Players need to be able to see if the names are taken
* Only one instance of a game can be in memory at a time

Business requirements as indicated by client:

* Develop game for multiple platforms
  + Windows
  + Mac
  + Linux
  + Mobile platforms
* Demonstrate which software development tools became utilized to perform business requirements
  + Cost benefit analysis

## [Design Constraints](#_2et92p0)

Using a web based distribution can have additional serialization and network latency overheads due to remote calls and RPCs. Also web distributions are potentially more complex and expensive environments overall. Also when buildings an application for mobile devices we must consider the differences in hardware, like screen size memory and battery life. As well as network bandwidth and security of the devices as well.

## [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 UML class diagram provided below shows us how classes are related to each other.

* The Program Driver and Singleton Tester classes alignment by the associated solid black arrow. The <<uses>> specifies that Program Driver employs Singleton Tester to demonstrate that only one instance is applicable during execution.
* The Game Service class and the Game class have a binary association, Identified with a solid black arrow. The multiplicity for this relationship further denoted by 0...\*, meaning zero to many.
* The Games Class and the teams Class also have a binary association, Identified with a solid black arrow. The multiplicity for this relationship further denoted by 0...\*, meaning zero to many.
* The team class and player class also have a binary association, Identified with a solid black arrow. The multiplicity for this relationship further denoted by 0...\*, meaning zero to many.
* The classes of game, team, and player all show an inheritance relationship with the class entity. That is denoted with an open arrow. This also means that the entity class is made up of the game, team, and player classes

**"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** | Advantages:   * Hardware is secure * Elevated levels of security * Flexible hosts * Apache servers allow for web code to run smooth   Disadvantages:   * Closed platforms (hard to push to other systems) * Cost (Expensive and proprietary) | Advantages:   * Low cost * Easy maintenance * Apache servers run basic web code * Reliability * Security * Self-updating   Disadvantages:   * High entry knowledge to use effectively | Advantages:   * Multiple applications at the same time * Host applications that others cant (ASP.net, access, MSSQL)   Disadvantages:   * Can slow over time and can cost more to maintain * Windows applications only | Advantages:   * Implement on mobile cloud * Scalability * Affordability * Hardware free * High speed * Real time backups for DR   Disadvantages:   * Security risks * Data management is difficult * Limited bandwidth |
| **Client Side** | Advantages:   * None apparent   Disadvantages:   * High Cost * Not user friendly * Multiple display needed | Advantages:   * Negligible cost compaired to others * Time to load is low   Disadvantages:   * Must be an expert to run this system | Advantages:   * Cheap * Support multiple screens natively * User friendly * Built in features to run game   Disadvantages:   * Form events are easy to detect * UNIX | Advantages:   * Cost is low to sustain * Client side services are fast to implement * Efficiency implementing JS, CSS, and QR   Disadvantages:   * NA |
| **Development Tools** | **Relevant programming languages for macOS include the following:**   * Python * C++ * Java     **The IDEs used on Macs include the following:**   * XCode * NetBeans * Eclipse   **Game development software used for Macs includes the following:**   * Unreal Engine * Google Cloud Platform * Unity * Adobe AIR * Scratch * Godot 2.0 * Gdevelop * Torque2D * Torque3D | **Relevant programming languages for Linux include the following:**   * Python * C * C++ * Fortran * Pascal * COBOL * Lisp   **The IDEs used on Linux include the following:**   * The Mono Project * Lazarus     **Game development software used for Linux includes the following:**   * Unity * Godot | **Relevant programming languages for Windows include the following:**   * C++ * C# * C * XML * HTML * CSS * PHP * JavaScript   **The IDEs used on Windows include the following:**   * Visual Studio * NetBeans * Eclipse   **Game development software used for Windows includes the following**:   * GameMaker * Studio 2 * Unity * Unreal Engine 4 | **Relevant programming languages for Mobile Devices include the following:**   * Objective-C * Swift * Python * C/C++ * Java * JavaScript   **The IDEs used on mobile devices include the following:**   * Android Studio * XCode * Visual Studio * IntelliJ * Xamarin   **Game development software used for mobile devices includes the following:**   * Unity * Unreal Engine * Solar2D, * Fusion 2. 5 |

## 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 using Windows to develop this application
2. **Operating Systems Architectures**: There are a few advantages to using Microsoft:
   * Flexibility within the architecture
     + Web
     + Client
     + Development tools
   * Steady learning curve
   * Can run multiple applications at once
   * Can host multiple servers
   * More developer controls
3. **Storage Management**: for storage management I would implement Azure
   * Cheap
   * Serverless storage
   * Easy to set up different environments
4. **Memory Management**: Windows has great memory management
   * Virtual address space for each 32 bit process
   * 4gb of memory viewable
   * Each process has 8 terabyte address space on 64-bit windows
   * All threads have access to visible address space
   * Cannot access another process
     1. Protects integrity of each process
   * Works with large data files such as websites
5. **Distributed Systems and Networks**:
   1. Windows utilizes distributed systems and networks called interposes communications (IPC).
      1. clipboards
      2. COM
      3. Data Copy
      4. DDE
      5. File Mapping
      6. Mailslots
      7. Pipes
      8. RPC
      9. Windows Sockets
   2. Can call functions remotely and uses data sharing between applications.
   3. Storing information in azure creates a robust experience and secures the data in the case of an outage or disaster.
6. **Security**: windows OS has robust security measures:
   * MFA
   * Specific Coding languages for developers to add MFA codes to their code to ID threats
   * Prefast Tool to find bugs quickly
   * Azure can quickly build IPsec Tunnels to the user