**Método de la ingeniería**

***Contexto problemático***

*La empresa desarrolladora de videojuegos Epic Games, reconocida por juegos como la saga gears of war, unreal tourment, paragon y fortnite, siendo el ultimo reconocido como el juego del año.* La empresa Epic Games está preocupada por la gran cantidad de recomendaciones que han dejado muchos de los jugadores de fortnite en el último par de meses, esta preocupación radica en algunos inconvenientes que los usuarios han sentido con el gameplay de sus partidas.

**Fase 1. Identificación del problema**

Las preocupaciones de la empresa se resumen en los siguientes problemas:

**Problema 1**

* + Los jugadores sienten la necesidad de ser emparejados con jugadores del mismo nivel o parecido, esta estadística es llamada **skill**.

**Problema 2**

* + Habiendo implementado un sistema de ranking que empareje jugadores con el **skill** parecido, se teme que la latencia varíe mucho, es decir que la desviación estándar del ping de los jugadores sea lejana.

**Problema 3**

* + No hay un sistema de ranking o modo de juego que solo empareje a jugadores de la misma plataforma.

**Problema 4**

* + No existen variantes del juego para celebrar ocasiones especiales. Se necesita desarrollar un modo de juego para celebrar san valentin.

**REQUERIMIENTOS FUNCIONALES**

|  |  |
| --- | --- |
| Nombre | R1 – Diseñar ranking igualado |
| Resumen | Se va a diseñar un ranking en donde se podrá categorizar a los jugadores según su destreza para el juego |
| Entrada | Jugador – Skill |
| Resultado | Ranking diseñado correctamente |

|  |  |
| --- | --- |
| Nombre | R2 – Realizar partidas en modo plataforma |
| Resumen | Se van a llevar a cabo partidas según la plataforma en la que esté jugando el usuario |
| Entrada | Jugador – Plataforma |
| Resultado | Partidas realizadas con exíto |

|  |  |
| --- | --- |
| Nombre | R3 – Crear variantes del juego para San Valentín |
| Resumen | Se va a crear una variante del juego para la fecha de San Valentín. Este modo se basa en el manejo de las armas recogidas del juego |
| Entrada |  |
| Resultado | Variante creada con satisfacción |

|  |  |
| --- | --- |
| Nombre | R4 – Rankear Jugador |
| Resumen | Se clasificara el jugador dentro del ranking de acuerdo a su latencia y su nivel de habilidad de juego. |
| Entrada | Jugador. |
| Resultado | Clasifica al jugador dentro de un ranking con jugadores que tengan una latencia cercana y un nivel de habilidad cercano. |

**Fase 2. Recopilación de la información**

Para la resolución de este problema es de primera necesidad conocer y contextualizar los siguientes conceptos:

**1.Ping**

Ping se utiliza como diagnóstico para asegurar que una computadora host (huésped) que el usuario está tratando de alcanzar está actualmente operativa. Ping funciona mediante el envío de un Internet Control Message Protocol (ICMP), petición de eco a una interfaz especificada en la red y espera una respuesta. Ping se puede utilizar para solucionar problemas de conectividad y determinar el tiempo de respuesta.

**2.Latencia**

En redes informáticas de datos la **latencia** es la suma de retardos temporales dentro de una red. Un retardo es producido por la demora en la propagación y transmisión de paquetes dentro de la red.

Es el tiempo exacto que tarda en transmitirse un paquete dentro de la red, **el tiempo que tardas en recibir un paquete del servidor**. Lo que mide es la inmediatez de la conexión, y si por ejemplo tenemos una latencia de 50 milisegundos significa que ese es el tiempo que tardan en llegar desde por ejemplo los servidores de Google hasta nuestro ordenador.

**3.Plataforma de videojuegos**

En informática, una **plataforma** es un sistema que sirve como base para hacer funcionar determinados módulos de hardware o de software con los que es compatible. Dicho sistema está definido por un estándar alrededor del cual se determina una arquitectura de hardware y una plataforma de software (incluyendo entornos de aplicaciones). Al definir plataformas se establecen lo tipos de arquitectura, sistema operativo, lenguaje de programación o interfaz de usuario compatibles.

En el caso de las **plataformas de videojuegos,** son los distintos tipos de dispositivo en los que se ejecutan los videojuegos. Los cuatro tipos de plataforma más populares son el PC, las videoconsolas, los dispositivos portátiles y las máquinas arcade.

**4.MOBA**

Una **arena de combate multijugador en línea** o **MOBA** (del inglés *multiplayer online battle arena*, también conocido como **estrategia de acción en tiempo real** o **ARTS** (del inglés *action real-time strategy*), es un subgénero de la [estrategia en tiempo real](https://es.wikipedia.org/wiki/Videojuego_de_estrategia_en_tiempo_real) (RTS), en el que dos equipos de jugadores compiten entre sí, con cada jugador controlando un solo personaje a través de una interfaz de estilo RTS, en un mapa usualmente simétrico, con el objetivo de destruir la base del oponente.**estadio de combate** o **campo de combate** en lugar de *arena de combate*.

**5.MMORPG**

Los **videojuegos de rol multijugador masivos en línea** o **MMORPG** (sigla en inglés de *massively multiplayer online role-playing game), son videojuegos de rol que permiten a miles de jugadores introducirse en un mundo virtual de forma simultánea a través de internet e interactuar entre ellos. Consisten, en un primer momento, en la creación de un personaje, del que el jugador puede elegir raza, profesión, armas, etc. Una vez creado el personaje el jugador puede introducirlo en el juego e ir aumentando niveles y experiencia en peleas JcJ (jugador contra jugador) o JcE (jugador contra entorno) o llevando a cabo diversas aventuras o completando misiones.*

**Fase 3. Búsqueda de soluciones creativas**

**Método de búsqueda de soluciones: Lluvia de ideas**

Se realizo una lluvia de ideas y se obtuvieron las siguientes nociones:

Problema 1:

* Recibir la skill como parametro definido sin aplicar ningún tipo de calculo:
  + Al recibir la **skill** como parámetro no tendríamos la necesidad de realizar cálculos, al crearse un jugador esta la tendría como un atributo ya definido.
* Calcular la **skill** a partir de las estadísticas de los jugadores. Estas estadísticas tendrían un valor cuantitativo dado por nosotros que va a generar un impacto en el cálculo del coeficiente de **skill** el cual representan la habilidad del jugador.
  + No recibir el **skill** como un parametro, si no, calcularlo, dandole un valor cuantitativo a las diferentes estadísticas de los jugadores por nosotros mismos.
* Teniendo la Skill calculada, se procede a clasificar los jugadores en algún tipo de estructura de datos. Lluvia de ideas sobre estructuras de datos:
  + Agregar los jugadores a una lista enlazada simple.
  + Agregar los jugadores a una cola de prioridad.
  + Usar una hastable para guardar los jugadores.

Problema 2:

**Estándar de clasificación:** El ping de cada jugador entrara como un dato ya definido, la propuesta es, dada un conjunto de jugadores los organizaremos en una hasTable, cada slot de las hashTable representara un rango de latencias entre los jugadores.

Problema 3:

**Estándar de clasificación:** La plataforma de cada jugador entrara como un dato ya definido.

La posible estructura de datos que vamos a utilizar para agrupar a los jugadores será:

* Usar una estructura de datos tipo cola para agrupar los jugadores.
  + Se crearían diferentes colas para diferentes partidas, una por partida.
* Usar una estructura de datos tipo cola (montículos)
  + Se crearía una única cola donde se puedan realizar saltos para agregar a los jugadores con el skill y el ping mas cercano.
* Agregar los jugadores a un arreglo para cada partida.
  + Agrupar a los jugadores que tengan un skill y un ping cercano en un arreglo de máximo 100 posiciones.
* Usar una estructura tipo lista enlazada de jugadores.
  + Cada jugador nuevo se agregará a una lista enlazada que luego será recorrida para hallar los jugadores con skill y ping cercanos.
* Usar una tabla hash para guardar los jugadores con una latencia y un skill cercano en un mismo slot con ayuda de las colisiones.

Problema 4:

* Modelar el inventario con una pila.
  + Utilizar una estructura de datos tipo pila para modelar el inventario de armas y así poder acceder solo a la ultima arma adquirida por el jugador.
* Modelar el inventario con un ArrayList.
  + Utilizar un ArrayList para agregar las armas siempre en la primera posición.
* Modelar el inventario con una lista enlazada.
  + Agregar a una lista enlaza las armas y modificarla cada vez que se elimina o agrega una nueva.

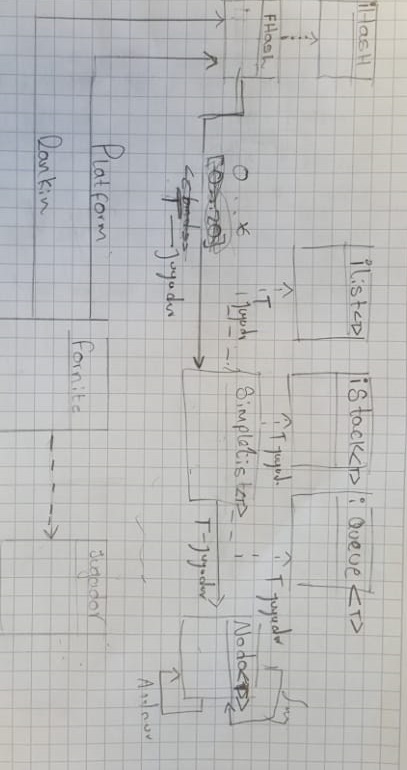
**Fase 4. Transición de la formulación de ideas a los diseños preliminares:**

Problema 1: Descarte de ideas no factibles

|  |  |
| --- | --- |
| Skill como parámetro | Al ser la skill pasada por parámetro no se estaría realmente calculando el nivel de habilidad del jugador. El valor de la skill depende de muchas variables, al recibirlo por parámetro no tendríamos la certeza de como se calculo ese valor. |

Problema 2 y 3: Descarte de ideas no factibles

|  |  |
| --- | --- |
| Estructura tipo cola | Usar una estructura de datos tipo cola para cada partida nos permitiría agregar a los jugadores con skill y ping similares en las mismas partidas, pero tendría una complejidad espacial muy alta y el tiempo de espera para comenzar una partida aumentaría considerablemente. El tiempo de espera y la complejidad aumentarían a la medida en que entren jugadores con ping o skill muy distantes. |
| Arreglo para partidas | Al utilizar un arreglo fijo de 100 posiciones el tiempo de espera para llenar los mismos y que no tengan posiciones vacías (nulas) aumentaría. Esto se vería reflejado en el tiempo de espera de emparejamiento para los jugadores. |
| Lista enlazada | Si se tiene una lista enlazada con todos los jugadores al momento de buscar los que tienen skill y ping cercanos la complejidad temporal seria O(n) siendo muy ineficiente esta manera de modelar el problema. |

Problema 4:

|  |  |
| --- | --- |
| **ArrayList** | Arraylist no es la estructura adecuada para modelar esta situación ya que su método agregar agrega en cualquier posición, lo cual no conviene en esta ocasión, si no, que cada vez que se encuentre una arma se agregue a la primera posición. |
| **Lista enlazada** | Una lista enlazada no esta la estructura adecuada para la situación, ya que cada vez que encontráramos un arma se tendría que aparte de agregarla, especificarle donde, y tendría que eliminar la otra, es decir, tocaría llamar otro método, aumentado la complejidad temporal y espacial. |

Borrador de diagrama de clases:

Fase 5 : Evaluación y selección de la mejor solución:

Criterios:

|  |  |
| --- | --- |
| * + - 1. Complejidad temporal.          1. 5 constante          2. 4 logarítmica          3. 3 n \* log n          4. 2 cuadrática          5. 1 factorial | * + - 1. Complejidad Espacial.          1. 5 constante          2. 4 logarítmica          3. 3 n \* log n          4. 2 cuadrática          5. 1 factorial |
| * + - 1. .Acoplamiento al problema          1. 3 acoplado          2. 2 poco acoplamiento          3. 1 no funciona en el problema | * + - 1. Complejidad de programación          1. 3 poco complejo          2. 2 complejo          3. 1 bastante complejo |

Estructura utilizada para rankear a los jugadores tanto por skill y lantencia como por Plataforma

|  |  |  |  |
| --- | --- | --- | --- |
| Estructura | 1 |  |  |
| Lista Enlazada |  |  |  |
| Cola De prioridad |  |  |  |
| Arreglo |  |  |  |
| HasTable |  |  |  |

Diseño de diagrama de clases de la solución

Diseño del diagrama de objetos

Diseño de casos de las pruebas unitarias