

UNIVERSIDAD CATÓLICA ANDRÉS BELLO

Extensión Guayana

Cátedra Institucional

Algoritmos y Programación III

**Informe sobre el desarrollo del proyecto “Minecraft”**

Presentado por:

Barreto, Luís C.I: 27506801

Gamboa, Diego C.I: 28215217

Morán, Karen C.I: 28161659

Profesor:

Ing. Jannelly Bello

Ciudad Guayana, julio de 2020

**Índice**

[**Capítulo I: EL PROBLEMA.** 4](#_Toc44707854)

[*Planteamiento del Problema 4*](#_Toc44707855)

[*Objetivo General 5*](#_Toc44707856)

[*Objetivos Específicos 5*](#_Toc44707857)

[**Capítulo II: ANÁLISIS.** 6](#_Toc44707858)

[*Requerimientos del Sistema 6*](#_Toc44707859)

[*Tablas Descriptivas de Casos de Uso. 8*](#_Toc44707860)

[*Tablas Descriptivas de Clases. 19*](#_Toc44707861)

[**Capítulo III: DISEÑO.** 32](#_Toc44707862)

[*Diagramas de Secuencia 32*](#_Toc44707863)

[*Diagramas de Colaboración 35*](#_Toc44707864)

[*Diseño de Interfaz 39*](#_Toc44707865)

**Introducción**

No cabe duda que la producción de videojuegos se ha convertido en un mercado sólido que representa una parte importante del ingreso económico de un país, como es el caso de los Estados Unidos; quien goza de un ingreso de aproximadamente 1.08 billones de dólares anualmente gracias a esta industria.

El extenso mercado que respalda esta área del ocio es una prueba del gran interés que suscita en la población. Dicha atención puesta sobre el campo puede ser utilizada de forma positiva para incentivar la recreación y el aprendizaje en las personas que participan en el uso de los videojuegos, así como también el de quienes los desarrollan.

Del análisis anterior surge la principal motivación del presente proyecto: desarrollar un videojuego que cumpla con los estándares del género independiente (conociendo el contexto de aprendices de los desarrolladores), y que además dicha meta proporcione a los participantes la iniciativa de indagar y aprender a usar las herramientas necesarias para lograrlo.

# **Capítulo I: EL PROBLEMA.**

## **Planteamiento del Problema**

Inspirado en el archiconocido juego de construcción, Minecraft, se requiere crear un videojuego en donde esta vez sí haya un objetivo principal: derrotar al EnderDragon antes que tus contrincantes.

La trama principal del juego debe consistir en una carrera de exploración, lucha y recolección de recursos contra los oponentes. Cada uno de los máximos 6 jugadores iniciarán en un bioma distinto, teniendo la capacidad de recorrerlo e ir abriéndose camino hacia el jefe de dicho mundo. Cuando derroten al jefe de su área, se les gratificará con una pieza de diamante. Luego, el poder de un ojo de Ender los teletransportará a un bioma distinto en donde también deberán derrotar al jefe correspondiente.

Para poder luchar contra el jefe final, el EnderDragon, los participantes deberán haber probado su valentía haciéndose de una armadura completa de diamante y armas del mismo material; es decir, tendrán que haber derrotado a los 6 jefes, uno por cada bioma. Dicho proceso se deberá ver dificultado por la lucha contra los mobs agresivos del mundo abierto y por las peleas entre oponentes que se crucen durante la exploración.

Si un jugador muere, el ojo de Ender lo resucitará sobre punto de inicio, pero dejando su inventario vacío a costa de esto. Para enmendar la pérdida de puntos de vida durante las batallas, el jugador podrá consumir los alimentos que recolectó o compró para sanarse.

En cuanto a la interfaz del menú, se requiere que el usuario sea capaz de registrarse, iniciar sesión, consultar datos de sí mismo y otros usuarios, así como también deberá poder consultar manuales del juego.

## **Objetivo General**

Desarrollar un videojuego inspirado en Minecraft con la tecnología Java, siguiendo los estándares de la Programación Orientada a Objetos.

## **Objetivos Específicos**

* Estructurar y planificar el software mediante el Lenguaje Unificado de Modelado (UML).
* Estudiar el IDE Netbeans, implementación de sockets, librerías Java y demás herramientas técnicas a utilizar durante el desarrollo del videojuego.
* Maquetar interfaces gráficas de usuario mediante la librería Swing de Java.
* Ejecutar el proceso iterativo de desarrollo y depuración del software mientras se realiza la adecuada documentación del mismo.

# **Capítulo II: ANÁLISIS.**

## **Requerimientos del Sistema**

El videojuego a desarrollar deberá cumplir con los siguientes requisitos:

**Sobre los datos.**

* Permitir el registro y posteriormente inicio de sesión de un usuario, haciendo uso de unas credenciales indicadas por él mismo.
* Permitir modificar y/o eliminar datos personales del usuario.
* Mostrar estadísticas de las partidas de los usuarios.

**Sobre la jugabilidad.**

* Poder explorar los biomas con la modalidad de “mundo abierto”.
* Ser capaz de atacar a monstruos, criaturas u otros jugadores.
* Tener la capacidad de recoger objetos del suelo o los que hayan soltado los mobs u oponentes asesinados.
* Consultar inventario, así como soltar o usar los objetos contenidos en él.
* Comerciar con un aldeano, es decir, poder vender o comprar cosas desde su inventario.

**Sobre el menú.**

* Presentar la posibilidad de consultar un manual del juego, así como información sobre los desarrolladores y las tecnologías utilizadas.



**Diagrama de Casos de Uso**

## **Tablas Descriptivas de Casos de Uso.**

**Administrar usuario.**

|  |  |
| --- | --- |
| Caso de uso | Registrar Usuario. |
| Actores | Usuario. |
| Propósito | Registrar en la base de datos, la información asociada a un nuevo usuario: username, e-mail, password, entre otros. |
| Precondiciones | * Ingresar un username no registrado. * Ingresar un e-mail válido. * Ingresar un password válido. * Confirmar su password. |
| Flujo Principal | De ser correctos todos los datos ingresados por el usuario, éstos se guardan en la base de datos. |
| Flujo alternativo | De faltar algún campo o ser inválido(s), se procede a mostrar un mensaje de error para que el usuario ingrese los datos correctamente. |

|  |  |
| --- | --- |
| Caso de uso | Iniciar Sesión. |
| Actores | Usuario. |
| Propósito | Cargar en el juego toda la información asociada a un usuario previamente registrado en la base de datos: username, estadísticas, entre otros. |
| Precondiciones | * Ingresar un username que se encuentre en la base de datos. * Ingresar la clave asociada con el mismo username. |
| Flujo Principal | De ser correctos todos los datos ingresados, se cargan en el juego todos los datos asociados a este usuario. |
| Flujo alternativo | De faltar algún campo o ser inválido(s), se procede a mostrar un mensaje de error para que el usuario ingrese los datos correctamente. |

**Administrar estadísticas.**

|  |  |
| --- | --- |
| Caso de uso | Consultar estadísticas. |
| Actores | Usuario. |
| Propósito | Mostrar en pantalla las estadísticas asociadas a un username ingresado: cantidad de partidas jugadas, ganadas, perdidas y abandonadas. |
| Precondiciones | * Ingresar un username que se encuentre en la base de datos. |
| Flujo Principal | De ser un username registrado, se imprimen en pantalla las estadísticas asociadas al mismo. |
| Flujo alternativo | De ser un username inválido, se procede a mostrar un mensaje de error que indique que dicho username no está registrado en la base de datos. |

|  |  |
| --- | --- |
| Caso de uso | Eliminar estadísticas. |
| Actores | Usuario. |
| Propósito | Borrar de la base de datos, las estadísticas asociadas al usuario que inició esta sesión. |
| Precondiciones | * Tener estadísticas guardadas en la base de datos. |
| Flujo Principal | De tener estadísticas guardadas en la base de datos, se eliminan dichos datos del mismo. |
| Flujo alternativo | De no tener estadísticas guardadas en la base de datos, se muestra un mensaje al usuario que indique que éste no tiene estadísticas que eliminar. |

**Administrar información del juego.**

|  |  |
| --- | --- |
| Caso de uso | Consultar reglas de juego y como jugar. |
| Actores | Usuario. |
| Propósito | Mostrar al usuario las reglas, los controles y como jugar. |
| Precondiciones | * El usuario debe hacer click en el botón “Ayuda”. |
| Flujo Principal | Imprimir en pantalla las instrucciones que se encuentran en el archivo correspondiente. |
| Flujo alternativo | Se muestra un mensaje de error al usuario y le permite regresar a la pantalla anterior. |

|  |  |
| --- | --- |
| Caso de uso | Consultar información de desarrollo de juego. |
| Actores | Usuario. |
| Propósito | Mostrar al usuario la información correspondiente al desarrollo de la aplicación: lenguaje de programación y librerías externas utilizadas en el desarrollo del juego, desarrolladores, versión actualizada. |
| Precondiciones | * El usuario debe hacer click en el botón “Acerca de”. |
| Flujo Principal | Imprimir en pantalla la información de desarrollo de juego que se encuentran en el archivo correspondiente. |
| Flujo alternativo | Se muestra un mensaje de error al usuario y le permite regresar a la pantalla anterior. |

**Administrar partida.**

|  |  |
| --- | --- |
| Caso de uso | Hostear Partida. |
| Actores | Usuario. |
| Propósito | Crear una partida nueva. |
| Precondiciones | * Luego de iniciar sesión, el usuario debe hacer click en el botón “Crear partida”. |
| Flujo Principal | Se crea un servidor y este usuario se une a dicho servidor para iniciar una nueva partida. |
| Flujo alternativo | Se muestra un mensaje de error al usuario y le permite regresar a la pantalla anterior. |

|  |  |
| --- | --- |
| Caso de uso | Unirse a Partida. |
| Actores | Usuario. |
| Propósito | Se une a un servidor existente para iniciar una partida. |
| Precondiciones | * Luego de iniciar sesión, el usuario debe hacer click en el botón “Unirse a partida”. * Ingresar el IP de un servidor válido. |
| Flujo Principal | En caso de ingresar un IP válido, este usuario se une al servidor proporcionado por él. |
| Flujo alternativo | En caso de ingresar un IP inválido o no poder unirse correctamente a la partida, se muestra un mensaje de error al usuario y le permite regresar a la pantalla anterior. |

**Administrar personaje.**

|  |  |
| --- | --- |
| Caso de uso | Seleccionar Género. |
| Actores | Usuario. |
| Propósito | El usuario elige el género de su personaje antes de comenzar una partida. |
| Precondiciones | * El usuario debe haber iniciado sesión y haberse unido a un servidor o partida. * En dicho servidor deben estar al menos 2 jugadores, hasta 6 máximo. |
| Flujo Principal | Se asigna un género al personaje del usuario, y dependiendo de su elección, se asignan características que lo identifique: cabello largo o corto, entre otros. |
| Flujo alternativo | En caso de no especificar el género, se muestra un mensaje al usuario para que llene este campo. |

|  |  |
| --- | --- |
| Caso de uso | Asignar nombre al personaje. |
| Actores | Usuario. |
| Propósito | El usuario asigna el nombre del personaje con el que jugará dicha partida. |
| Precondiciones | * El usuario debe haber iniciado sesión y haberse unido a un servidor o partida. * En dicho servidor deben estar al menos 2 jugadores, hasta 6 máximo. * El nombre asignado debe tener como mínimo 4 caracteres, y máximo 16. * Otro jugador no puede haber escogido ese mismo nombre. |
| Flujo Principal | Se asigna el nombre al personaje del usuario y bloquea este nombre para los otros jugadores. |
| Flujo alternativo | En caso de ingresar un nombre con caracteres fuera de los parámetros establecidos, se muestra un mensaje de error. En caso de escribir un nombre ya elegido por otro jugador, se muestra un mensaje que lo indique. |

|  |  |
| --- | --- |
| Caso de uso | Seleccionar color de ropa y cabello. |
| Actores | Usuario. |
| Propósito | El usuario elige el color de ropa y de cabello que tendrá su personaje en dicha partida. |
| Precondiciones | * El usuario debe haber iniciado sesión y haberse unido a un servidor o partida. * En dicho servidor deben estar al menos 2 jugadores, hasta 6 máximo. |
| Flujo Principal | Se asigna un género al personaje del usuario, y dependiendo de su elección, se asignan características que lo identifique: cabello largo o corto, entre otros. |
| Flujo alternativo | En caso de no especificar el género, se muestra un mensaje al usuario para que llene este campo. |

**Jugar.**

|  |  |
| --- | --- |
| Caso de uso | Jugar. |
| Actores | Usuario. |
| Propósito | Cuando el usuario se encuentra en este caso de uso, podrá realizar cualquiera de las siguientes acciones dentro de la partida en curso: explorar bioma, ir a la zona de comercio, consultar inventario. |
| Precondiciones | * El usuario debe haber iniciado sesión, haberse unido a un servidor o partida y haber personalizado su personaje. * En dicho servidor deben estar al menos 2 jugadores, hasta 6 máximo. |
| Flujo Principal | A cada jugador dentro del servidor se le asigna un bioma aleatorio el cual será su punto de partida, luego, cada uno de ellos dentro de ese bioma podrá realizar las acciones anteriormente descritas. |
| Flujo alternativo | En caso de no lograr iniciar la partida, se muestra un mensaje de error al usuario y posteriormente, éste regresara a la pantalla principal. |

**Ir a la zona de comercio.**

|  |  |
| --- | --- |
| Caso de uso | Ir a la zona de comercio. |
| Actores | Usuario. |
| Propósito | El usuario podrá consultar los objetos disponibles en la zona de comercio que atienden los aldeanos que se encuentran en cada uno de los biomas que conforman el juego. |
| Precondiciones | * El usuario debe haber iniciado una partida y estar vivo. |
| Flujo Principal | Al jugador entrar en la zona de comercio, se mostrarán en pantalla los objetos disponibles con su respectivo valor en esmeraldas y le permitirá as opciones de compra y venta de objetos de batalla o comida. |
| Flujo alternativo | En caso de no poder acceder a la zona de comercio, se muestra un mensaje de error al usuario. |

|  |  |
| --- | --- |
| Caso de uso | Comprar objeto (arma / comida). |
| Actores | Usuario. |
| Propósito | El usuario podrá comprar cualquier objeto de batalla o alimentos que se encuentren disponibles en la zona de comercio. |
| Precondiciones | * El usuario debe estar en la zona de comercio. * Tener la cantidad de esmeraldas suficientes para la compra del objeto de su preferencia. * En caso de tratarse de un objeto de batalla, verificar que este objeto no esté dentro del inventario del jugador |
| Flujo Principal | Se descontarán la cantidad de esmeraldas que cuesta el objeto seleccionado del monedero del jugador en cuestión, y se le cargará en su inventario el objeto comprado. |
| Flujo alternativo | En caso de no contar con la cantidad suficiente de esmeraldas o que un objeto de batalla esté en el inventario del jugador, se muestra un mensaje de error al usuario indicando que ocurre. |

|  |  |
| --- | --- |
| Caso de uso | Vender objeto (arma / comida). |
| Actores | Usuario. |
| Propósito | El usuario podrá vender cualquier objeto de batalla o alimentos que se encuentren disponibles en su inventario (excepto las piezas de la armadura especial y la espada de diamante) para obtener su valor en esmeraldas indicado en la zona de comercio. |
| Precondiciones | * El usuario debe estar en la zona de comercio. * Tener disponible en su inventario el objeto de batalla o alimento que desea vender. |
| Flujo Principal | Se eliminará del inventario del jugador el objeto que ha sido vendido y se añadirá a su monedero el valor de dicho objeto en esmeraldas. |
| Flujo alternativo | En caso de no contar con el objeto que desea vender dentro de su inventario, se muestra un mensaje de error al usuario indicando que ocurre. |

**Consultar inventario.**

|  |  |
| --- | --- |
| Caso de uso | Consultar inventario. |
| Actores | Usuario. |
| Propósito | El usuario podrá consultar y ver los objetos de batalla y el alimento que está dentro de su inventario, de los cuales podrá disponer en cualquier momento de la partida. |
| Precondiciones | * Estar vivo. |
| Flujo Principal | Se muestran en pantalla todos los objetos que cuenta en su inventario y lo que puede hacer con cada uno: comer alimento, portar o dejar de portar un objeto de batalla. |
| Flujo alternativo | En caso de no poder consultar el inventario se muestra un mensaje de error al usuario. |

|  |  |
| --- | --- |
| Caso de uso | Comer alimento. |
| Actores | Usuario. |
| Propósito | Permitir al usuario comer algún alimento que se encuentre disponible dentro de su inventario. |
| Precondiciones | * Estar dentro del caso de uso “Consultar inventario”. * Tener disponible en su inventario el alimento que se desea comer. |
| Flujo Principal | Se elimina el alimento comido dentro del inventario y se suma la vida que aportó dicho alimento a la vida total del jugador que lo consumió. |
| Flujo alternativo | En caso de no contar con el alimento elegido, se muestra un mensaje de error al usuario. |

|  |  |
| --- | --- |
| Caso de uso | Portar objeto de batalla. |
| Actores | Usuario. |
| Propósito | Equipar al personaje de un jugador con un objeto de batalla (armadura / arma) que esté disponible en su inventario. |
| Precondiciones | * Estar dentro del caso de uso “Consultar inventario”. * Tener disponible en su inventario, el objeto de batalla que desea portar. |
| Flujo Principal | En el personaje del jugador en cuestión, se equipa el objeto de batalla elegido y se coloca el estado de “equipado” en dicho objeto dentro del inventario. |
| Flujo alternativo | En caso de no contar con el objeto de batalla elegido en el inventario, se muestra un mensaje de error al usuario. |

|  |  |
| --- | --- |
| Caso de uso | Dejar de portar objeto de batalla. |
| Actores | Usuario. |
| Propósito | Quitarle al personaje de un jugador un objeto de batalla (armadura / arma) que esté equipado. |
| Precondiciones | * Estar dentro del caso de uso “Consultar inventario”. * Tener equipado, el objeto de batalla que desea dejar de portar. |
| Flujo Principal | En el personaje del jugador en cuestión, se elimina el objeto de batalla elegido y se coloca el estado de “disponible” en dicho objeto dentro del inventario. |
| Flujo alternativo | En caso de no estar equipado con el objeto de batalla elegido en el inventario, se muestra un mensaje de error al usuario. |

**Explorar Bioma.**

|  |  |
| --- | --- |
| Caso de uso | Explorar Bioma. |
| Actores | Usuario. |
| Propósito | Permitir al jugador moverse por un bioma y tener múltiples acciones dentro del misma. |
| Precondiciones | * Estar vivo. |
| Flujo Principal | El jugador puede moverse de cualquier forma por todo el bioma en donde se localice. |
| Flujo alternativo | En caso de morir o algún otro error, muestra un mensaje a usuario que indique su estado y le permite al usuario regresar a la pantalla principal. |

|  |  |
| --- | --- |
| Caso de uso | Equipar objeto al inventario. |
| Actores | Usuario. |
| Propósito | Permitir al jugador equipar al inventario algún alimento que encuentre en un bioma determinado. |
| Precondiciones | * Estar vivo. * Encontrar en un bioma un objeto de tipo alimento: carne, baya, pera, manzana, sandía, zanahoria y papa. |
| Flujo Principal | Al jugador tocar un alimento, éste será agregado automáticamente a su inventario y será eliminado del bioma donde se encuentra. |
| Flujo alternativo | En caso de no poder equipar el alimento, se muestra un mensaje de error al usuario y se mantiene el alimento donde está. |

**Atacar.**

|  |  |
| --- | --- |
| Caso de uso | Atacar Monstruo. |
| Actores | Usuario. |
| Propósito | Permitir al jugador atacar a un monstruo dentro de la arena de combate. |
| Precondiciones | * Estar vivo. * Estar dentro de la arena de combate especial con el Jefe (Monstruo) que custodia el bioma que estaba siendo explorado por un jugador. * El jugador debe presionar la tecla correspondiente a la acción de “atacar”. |
| Flujo Principal | Cada vez que el jugador presione la tecla de atacar y esté a una distancia prudente del Jefe (Monstruo), este último será golpeado y se le restará una determinada cantidad de puntos de vida a su vida total, de acuerdo al tipo de arma que porte el jugador. |
| Flujo alternativo | En caso de no poder atacar al oponente, mostrar un mensaje de error al usuario. |

|  |  |
| --- | --- |
| Caso de uso | Atacar animal. |
| Actores | Usuario. |
| Propósito | Permitir al jugador atacar a una criatura pacífica que se encuentre dentro del bioma donde se encuentre el mismo. |
| Precondiciones | * Estar vivo * El jugador debe presionar la tecla correspondiente a la acción de “atacar”. |
| Flujo Principal | Cada vez que el jugador presione la tecla de atacar y esté a una distancia prudente de un animal, este último será golpeado y se le restará una determinada cantidad de puntos de vida a su vida total, de acuerdo al tipo de arma que porte el jugador. |
| Flujo alternativo | En caso de no poder atacar al oponente, mostrar un mensaje de error al usuario. |

|  |  |
| --- | --- |
| Caso de uso | Atacar a otro jugador. |
| Actores | Usuario. |
| Propósito | Permitir al jugador atacar a otro jugador con el que se encontró durante el transcurso de una partida. |
| Precondiciones | * Estar vivo. * Encontrarse con otro jugador. * Estar dentro de la arena de combate con el jugador encontrado. * El jugador debe presionar la tecla correspondiente a la acción de “atacar”. |
| Flujo Principal | Cada vez que el jugador presione la tecla de atacar y esté a una distancia prudente del otro jugador, este último será golpeado y se le restará una determinada cantidad de puntos de vida a su vida total, de acuerdo al tipo de arma que porte el jugador. |
| Flujo alternativo | En caso de no poder atacar al oponente, mostrar un mensaje de error al usuario. |

|  |  |
| --- | --- |
| Caso de uso | Matar. |
| Actores | Usuario. |
| Propósito | Matar a un oponente luego de una serie de ataques realizados por el jugador. |
| Precondiciones | * Estar vivo. * Atacar a un oponente. * La vida del oponente debe llegar a 0 tras el último ataque |
| Flujo Principal | Cuando la vida del oponente llega a 0, éste automáticamente muere y deja esparcido en el suelo o arena de combate todo su inventario, o en el caso de os Jefes, un cofre con una pieza de la armadura especial lo la espada de diamante. |
| Flujo alternativo | En caso de ser un mob inmortal (aldeanos), mostrar un mensaje que le indique al jugador la situación. |



**Diagrama de Clases**

## **Tablas Descriptivas de Clases.**

|  |  |
| --- | --- |
| Clase | Mob |
| Es una clase abstracta que servirá para todos los personajes que no sean el jugador. |
| Atributos | #descripción, es un string que sirve para guardar que tipo de mob es.  #Vida, es un int, utilizado para saber la cantidad de vida del mob.  #ubicación, es un objeto de la clase Ubicación, sirve principalmente para conocer donde se encuentra en los distintos mapas.  #velocidad, es un double, sirve para llevar constancia de la velocidad que lleve en cualquier momento. |
| Métodos | +Mover, devuelve un bool, sirve para trasladar a el mob de un lugar a otro, dentro del mismo bioma donde se encuentre. |

|  |  |
| --- | --- |
| Clase | Pasivo |
| Representa a cualquier mob pacifico, es decir, que no atacara al jugador. |
| Atributos | #puedeMorir, devuelve un bool el cual indicara si puede morir o no. |
| Métodos | +hacerRuido(), hace que el mob genere un sonido. |

|  |  |
| --- | --- |
| Clase | Aldeano |
| Mobs que comerciaran con el jugador. |
| Atributos | -inventario, de la clase Inventario, guarda el “Stock” del aldeano.  -listaPrecios, es una lista paralela a su inventario, donde se indican los precios. |
| Métodos | +Comerciar(Jugador), comerciara con el jugador que se le pase indique. |

|  |  |
| --- | --- |
| Clase | Animal |
| Representa a todos los animales. |
| Atributos | #CantCuracionCarne, devuelve un double, e indica cuanto recuperaras al consumir la carne de ese animal. |
| Métodos | +Huir(Jugador), devolverá un bool, el cual le indicara si debe correr del jugador. |

|  |  |
| --- | --- |
| Clase | Vaca |
| Representa a las vacas del juego. |
| Atributos | No tiene. |
| Métodos | No tiene. |

|  |  |
| --- | --- |
| Clase | Oveja |
| Representa a las ovejas. |
| Atributos | No tiene. |
| Métodos | No tiene. |

|  |  |
| --- | --- |
| Clase | Cerdo |
| Representa a los cerdos |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene atributos propios. |

|  |  |
| --- | --- |
| Clase | Conejo |
| Representa a los conejos. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene atributos propios. |

|  |  |
| --- | --- |
| Clase | Pollo |
| Representa a los pollos. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene atributos propios. |

|  |  |
| --- | --- |
| Clase | Pescado |
| Representa a los pescados |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene atributos propios. |

|  |  |
| --- | --- |
| Clase | Mounstro |
| Representa a todos los mobs agresivos, es decir los que intentaran atacar al jugador. |
| Atributos | #esJefe, es un booleano, que indica si es un jefe o no.  #dificultad, es un int, que representa el nivel de dificultad.  #cantAtaque, es un double, que representa cuanto daño pueden infligir.  #descripcion, es un string, que describe a los monstruos. |
| Métodos | +acercarse(Jugador), sirve para acerca al monstruo al jugador  +atacar(Jugador), sirve para atacar al jugador de forma básica. |

|  |  |
| --- | --- |
| Clase | EnderDragon |
| Representa al jefe principal del juego, el EnderDragon. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene atributos propios. |

|  |  |
| --- | --- |
| Clase | Zombie |
| Representa a los zombies del juego. |
| Atributos | No tiene atributos propios. |
| Métodos | +envenenar(Jugador), es el ataque especial del zombie, envenenara al jugador. |

|  |  |
| --- | --- |
| Clase | Esqueleto |
| Representa a los esqueletos. |
| Atributos | No tiene atributos propios. |
| Métodos | +lanzarFlecha(Jugador), es el ataque especial del esqueleto, lanza una flecha al jugador. |

|  |  |
| --- | --- |
| Clase | Araña |
| Representa a las arañas. |
| Atributos | No tiene atributos propios. |
| Métodos | +atrapar(Jugador), es el ataque especial de la araña, atrapa al jugador. |

|  |  |
| --- | --- |
| Clase | EnderMan |
| Representa al EnderMan |
| Atributos | No tiene atributos propios. |
| Métodos | +teletransportarse(), hace que se teletransporte a un lugar aleatorio en el bioma |

|  |  |
| --- | --- |
| Clase | Creeper |
| Representa al Creeper |
| Atributos | No tiene atributos propios. |
| Métodos | +explotar(), es el ataque especial del Creeper, explotara cerca del jugador. |

|  |  |
| --- | --- |
| Clase | CerdoZombie |
| Representa a los cerdoZombie. |
| Atributos | No tiene atributos propios. |
| Métodos | +quemar(Jugador), es el ataque especial del CerdoZombie, quemara al jugador cuando lo toque. |

|  |  |
| --- | --- |
| Clase | Bioma |
| Representa los 6 distintos biomas en el juego |
| Atributos | #descripcion: String, da una breve descripción al bioma.  #aldeano: Aldeano, cada bioma tiene un aldeano.  #tipoMonstruo: Monstruo, cada bioma cuenta con un tipo de mounstro único del lugar.  #tipoAnimal: Animal cada bioma cuenta con un tipo de animal único del lugar.  #tipoCosecha: Cosecha cada bioma cuenta con un tipo de vegetal o hortaliza único del lugar. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Bosque |
| Representa los bosques. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Playa |
| Representa las playas. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Cueva |
| Representa las Cuevas. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Pradera |
| Representa la pradera. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Nether |
| Representa el nether. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Ender |
| Representa el Ender. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Habitación |
| Representa las habitaciones del juego, es decir, cualquier escenario, como el mundo abierto o las salas de batallas. |
| Atributos | #descripcion, es un string que describe a la habitación.  #bioma: es un objeto de clase Bioma, para así asignarle un bioma a cada habitación.  #ancho: es un double, que representa el largo de la habitación.  #alto: es un double, que representa el alto de la habitación. |
| Clase | MundoAbierto |
| Representa el mundo abierto, donde el jugador se mueva libremente y sea capaz de explorar. |
| Atributos | -numJugador, es un int que representa el mundo en el que va a aparecer. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | SalaCombate |
| Representa las diversas salas para combatir como, salas de jefes o cuando se enfrente con a otros jugadores |
| Atributos | -nombre: es un string que guarda el nombre de la sala  -arrLuchadores: es un arreglo de dos jugadores, donde se guardaran máximo 2 jugadores. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Objeto |
| Representa todos los objetos que están en el juego |
| Atributos | #nombre, es un string que sirve para guardar el nombre del objeto  #tirado, es un booleano qué dice si el objeto esta tirado o no. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Pocion |
| Representa las pociones o hechizos del juego. |
| Atributos | -hechizo, es un string que guarda el nombre del hechizo.  -factorHechizo: es un double, representa que tan fuerte será el hechizo. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | ObjetoBatalla |
| Representa cualquier objeto que se pueda usar en combate. |
| Atributos | #descripcion: es un string donde se guarda la descripción de el objeto de batalla.  #material: es un objeto de clase Material, que indicara de qué material esté hecho.  #usoRestante: es un double, que guarda cuantos usos le quedan. |
| Métodos | +romperse(), sirve para romper el objeto en caso de que su usoRestante llegue a 0. |

|  |  |
| --- | --- |
| Clase | Arma |
| Es la clase para todas las armas. |
| Atributos | -cantAtaque, es un doublé que guarda el daño que hará el arma. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | PiezaArmadura |
| Es la clase para todas las piezas de armadura. |
| Atributos | -cantProteccion, es un doublé que guarda la capacidad de defensa que tiene las piezas de armadura. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Vestuario |
| Representa el vestuario, en general que se puede poner el jugador. |
| Atributos | -colorPiel, es de clase Color, para guardar el color de piel del jugador.  -colorRopa, es de clase Color, y sirve para guardar el color de la ropa del jugador.  -armadura, es un vector de PiezaArmadura de 4 posiciones, que representa las 4 piezas que se puede poner el jugador. |
| Clase | Alimento |
| Representa todos los alimentos. |
| Atributos | #cantCuracion es un double, que representa la cantidad de vida que recuperara esa comida. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Carne |
| Representa todas las carnes. |
| Atributos | -tiempoCaducidad es una variable interval, es decir, que guarda intervalos de tiempo, y representa el tiempo de caducidad de ese alimento.  -caducidad es un bool, que indica si ya está caducado.  -tipo es un objeto de clase animal, que indica que tipo de carne. |
| Métodos | No tiene métodos. |

|  |  |
| --- | --- |
| Clase | Cosecha |
| Representa todas las hortalizas, verduras y frutas que se pueden consumir. |
| Atributos | #estaHechizado, es un bool que indica si esta hechizado. |
| Métodos | -hechizar(Pocion), es una funcio que devolverá un bool si logro ser hechizado. |

|  |  |
| --- | --- |
| Clase | Papa |
| Representa a las papas. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos propios. |

|  |  |
| --- | --- |
| Clase | Zanahoria |
| Representa a las zanahorias. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos propios. |

|  |  |
| --- | --- |
| Clase | Manzana |
| Representa a las manzanas. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos propios. |

|  |  |
| --- | --- |
| Clase | Pera |
| Representa las peras. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos propios. |

|  |  |
| --- | --- |
| Clase | Sandia |
| Representa las sandias. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos propios. |

|  |  |
| --- | --- |
| Clase | Bayas |
| Representa las bayas. |
| Atributos | No tiene atributos propios. |
| Métodos | No tiene métodos propios. |

|  |  |
| --- | --- |
| Clase | Jugador |
| Representa al jugador, es decir, los personajes que se pueden controlar. |
| Atributos | -nombre: es un string que guarda el nombre de usuario.  -vida: es un double, que guarda la cantidad de vida que tiene.  -vestuario: es un objeto de clase Vestuario, que sirve para guardar el vestuario guardado por el usuario.  -ubicacion: es un objeto de clase Ubicación, que guarda las direcciones en las que se encuentra el jugador.  -cantEsmeraldas: es un int que guarda la cantidad de esmeraldas que posee actualmente el jugador.  -inventario: es un objeto de clase Inventario, que guarda todo el inventario del usuario.  -jefesGanados: es un vector de 6 posiciones de int, que se ira llenando cuando derrotes a los jefes de cada área. |
| Métodos | +mover(Ubicacion), devuelve un bool que indica si se logró mover a esa coordenada.  +comer(Alimento), permite al jugador alimentarse y curar vida  +atacar(Jugador,Arma), devuelve un bool que dirá si logro atacar a otro jugador.  +atacar(Mob, Arma): devuelve un bool que dirá si logro atacar al mob.  +usar(ObjetoBatalla), es para usar un objeto de batalla.  +dejar(ObjetoBatalla), es para poder botar un objeto al mundo abierto y quitarlo de tu inventario |

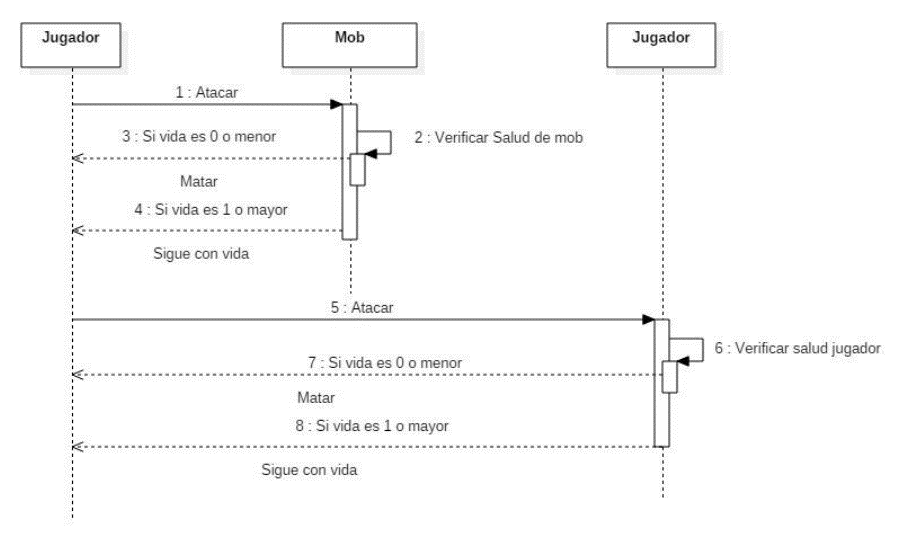
|  |  |
| --- | --- |
| Clase | Inventario |
| Representa el inventario y sirve para gestionarlo. |
| Atributos | -listaObjetos, es un ArrayList de Objeto, para guardar todo el inventario.  -cantPorObjeto, es un int que guarda la cantidad de objetos. |
| Métodos | +agregar(Objeto) es un método que devuelve un bool, que indicara si se pudo agregar el objeto al inventario.  +sacar(Objeto), sirve para eliminarlo de la lista de objetos .  +verInventario(), sirve para visualizar el inventario. |

|  |  |
| --- | --- |
| Clase | Ubicación |
| Representa la ubicación de un personaje del juego. |
| Atributos | -habitación, es un int que va de 1 a 6, que representa en que habitación se encuentra.  -posX: es un double que guarda la coordenada X.  -posY: es un double que guarda la coordenada Y. |
| Métodos | No tiene. |

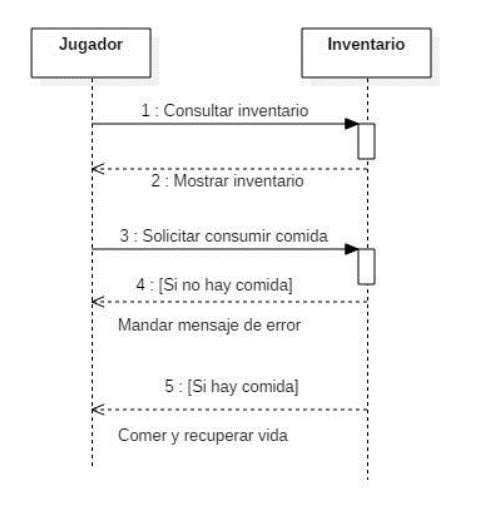
# **Capítulo III: DISEÑO.**

## **Diagramas de Secuencia**

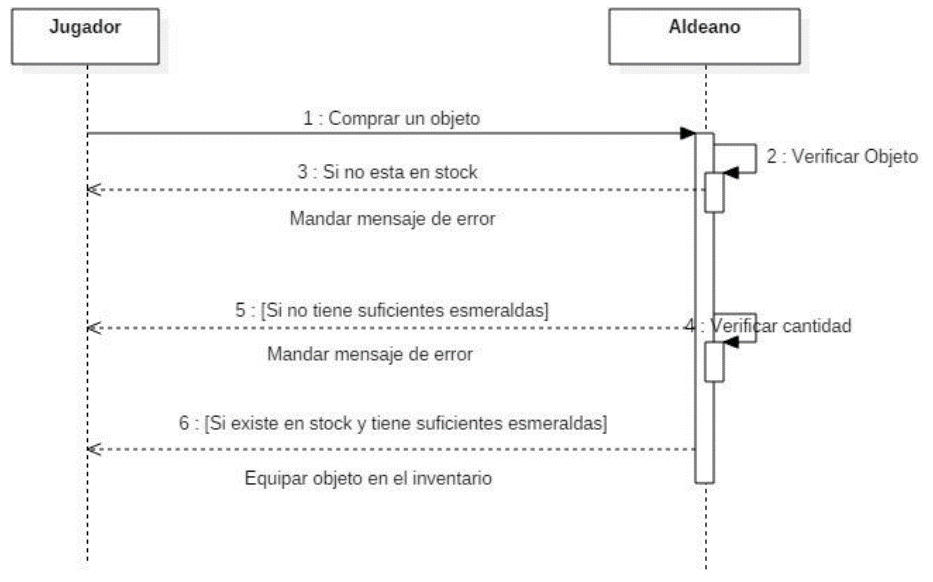
**Atacar.**



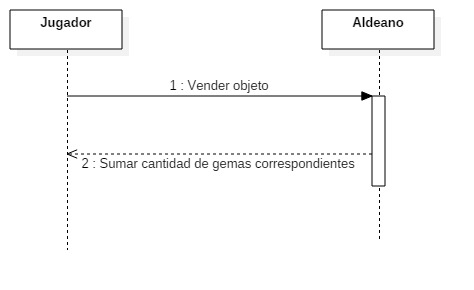
**Comer.**

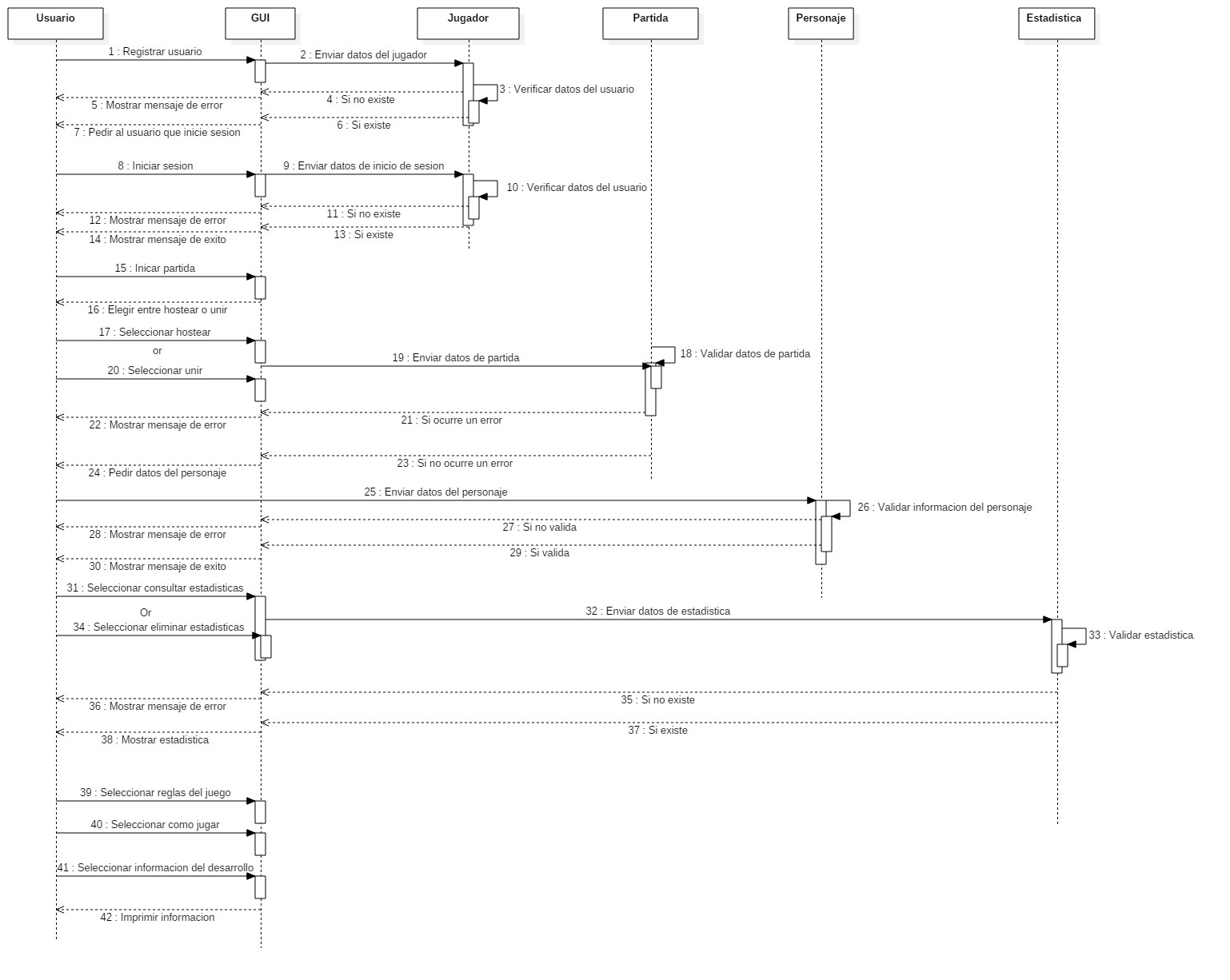


**Comerciar.**



**Vender.**



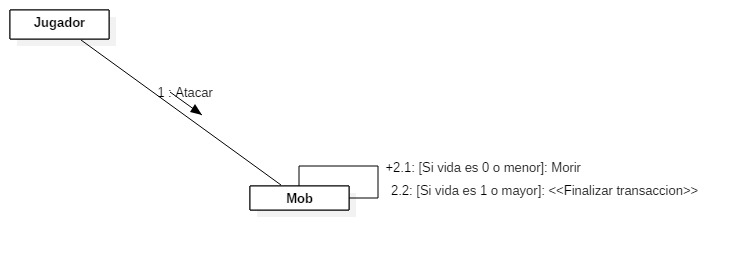
**Menú.**

## **Diagramas de Colaboración**

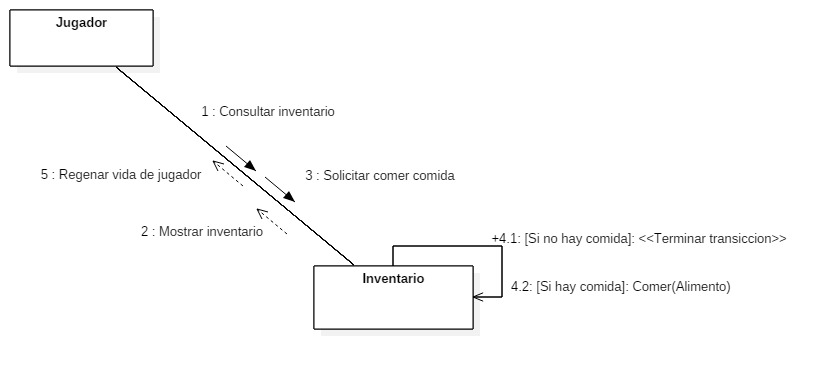
**Atacar Jugador.**



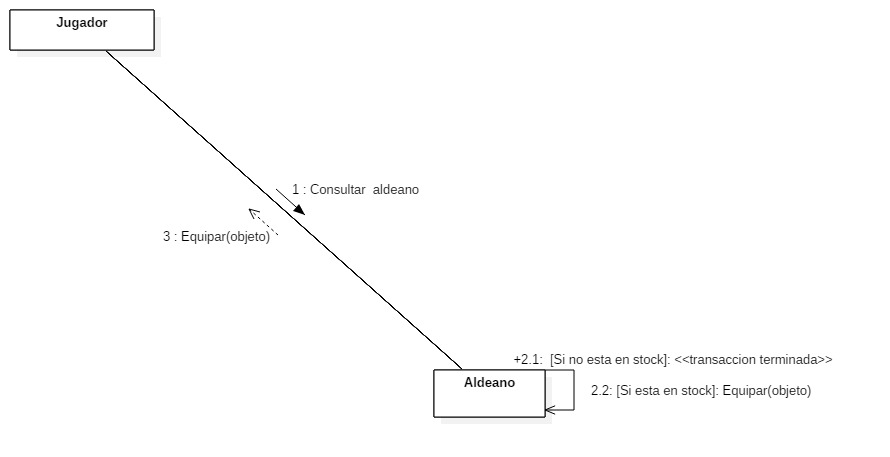
**Atacar Mob.**



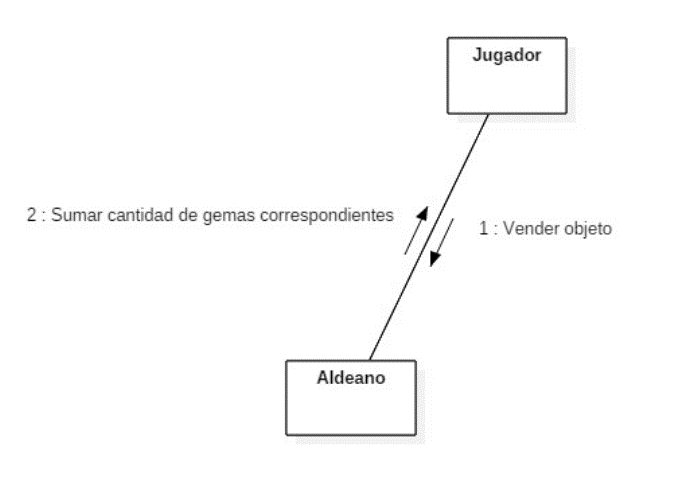
**Comer.**



**Comerciar.**

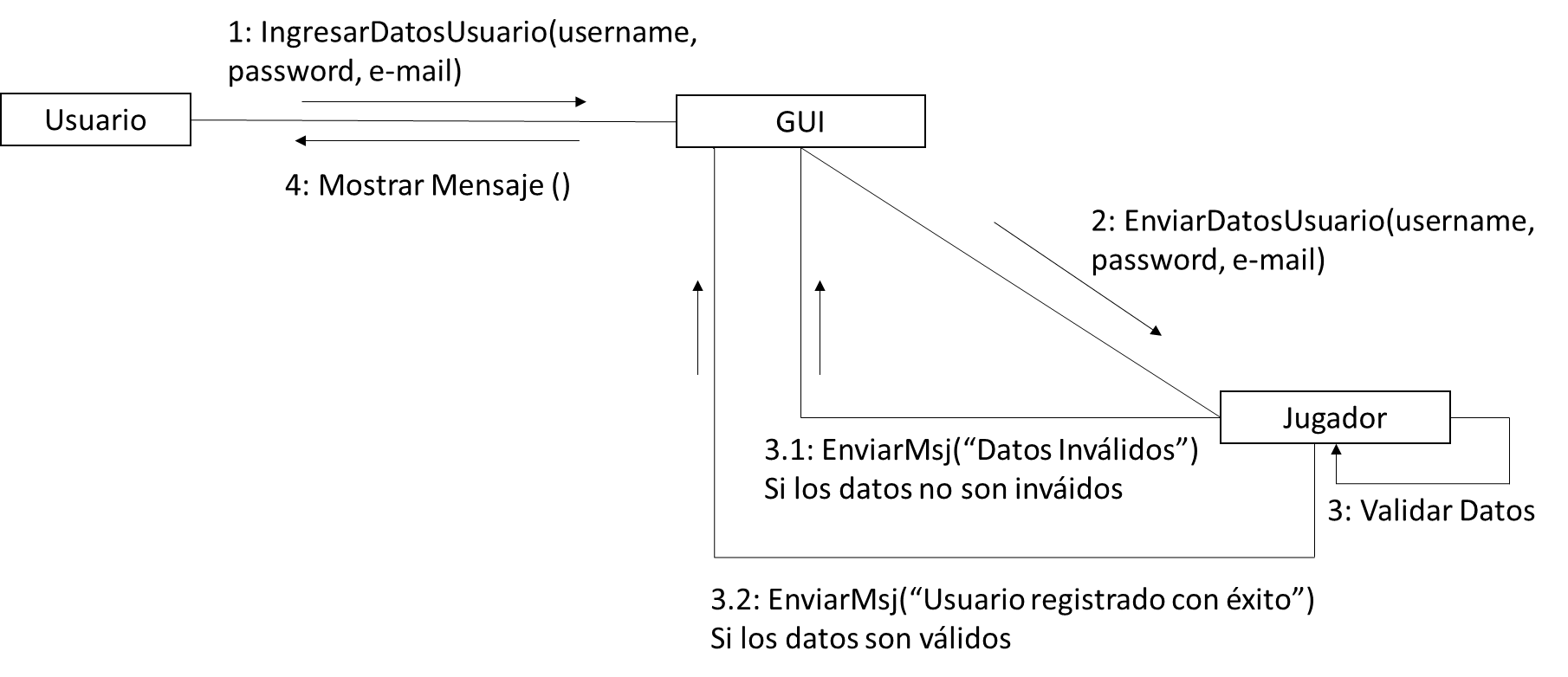


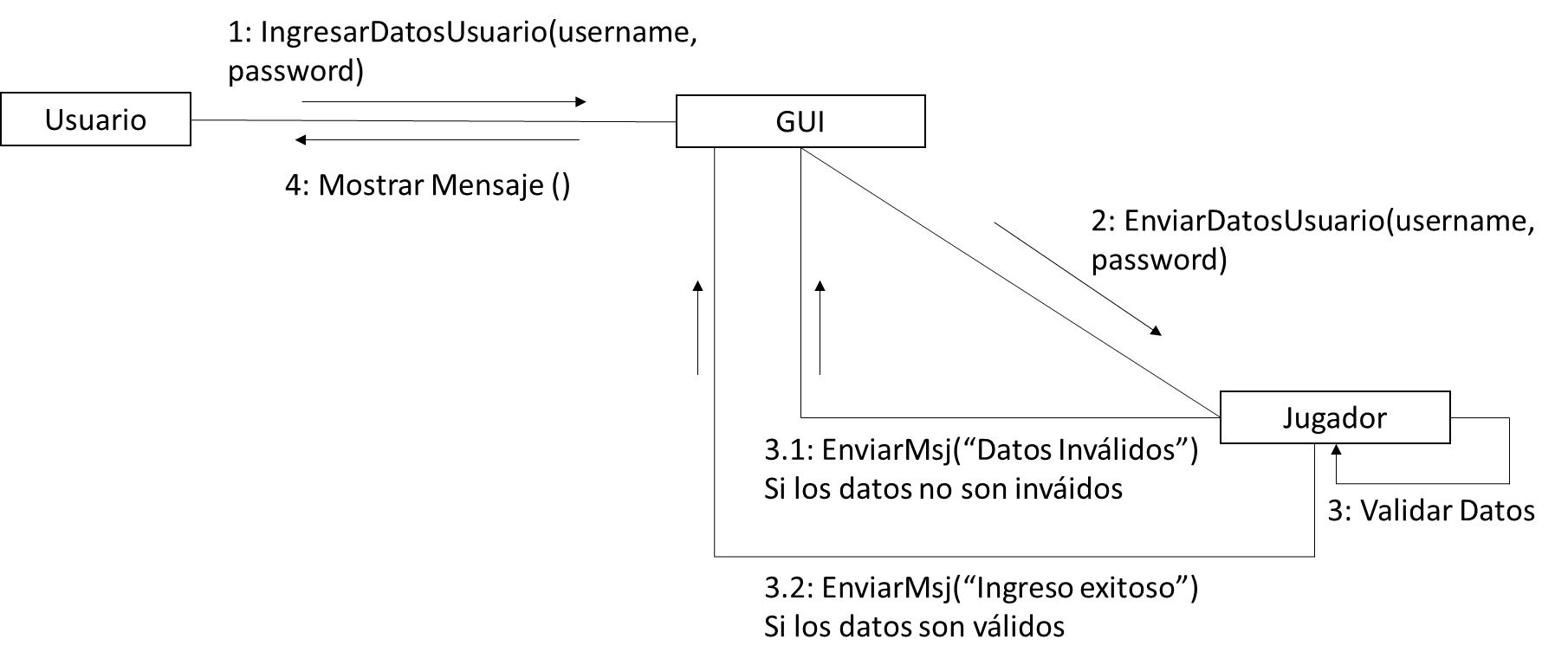
**Vender.**



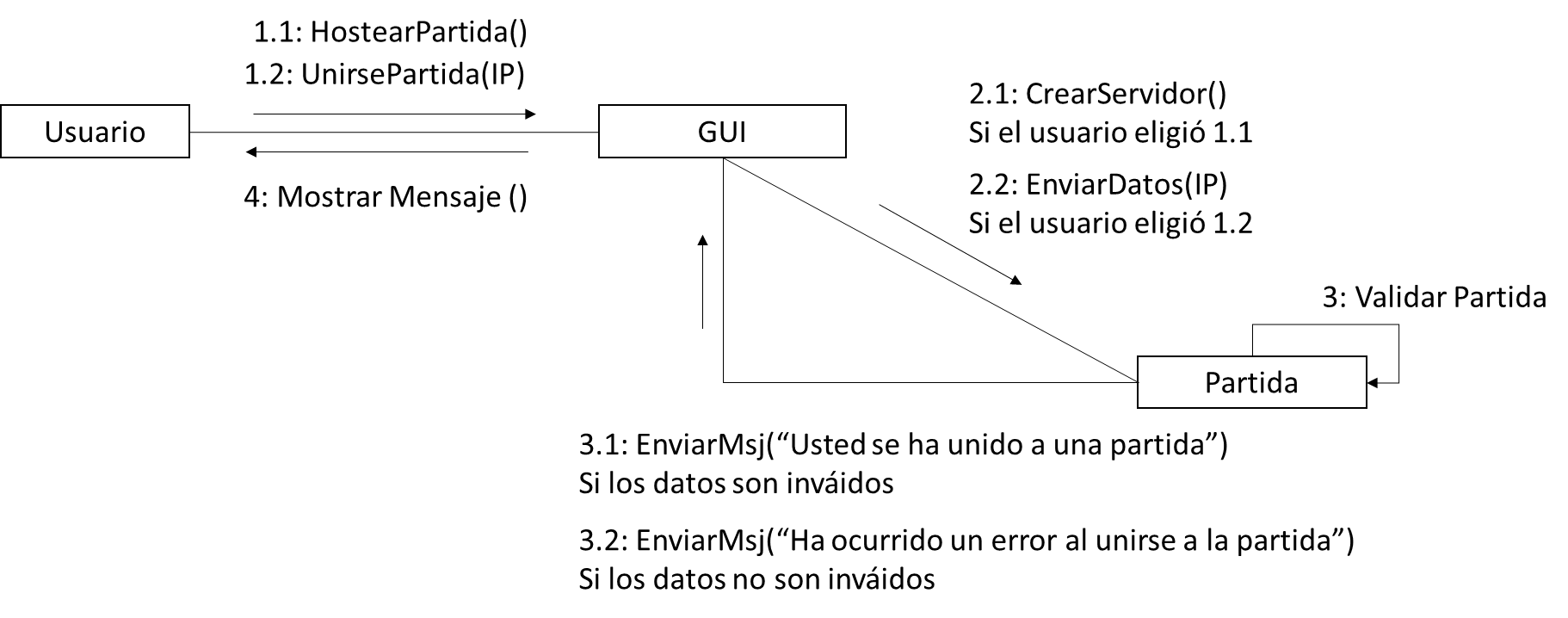
**Menú.**

***Registro de usuario.***

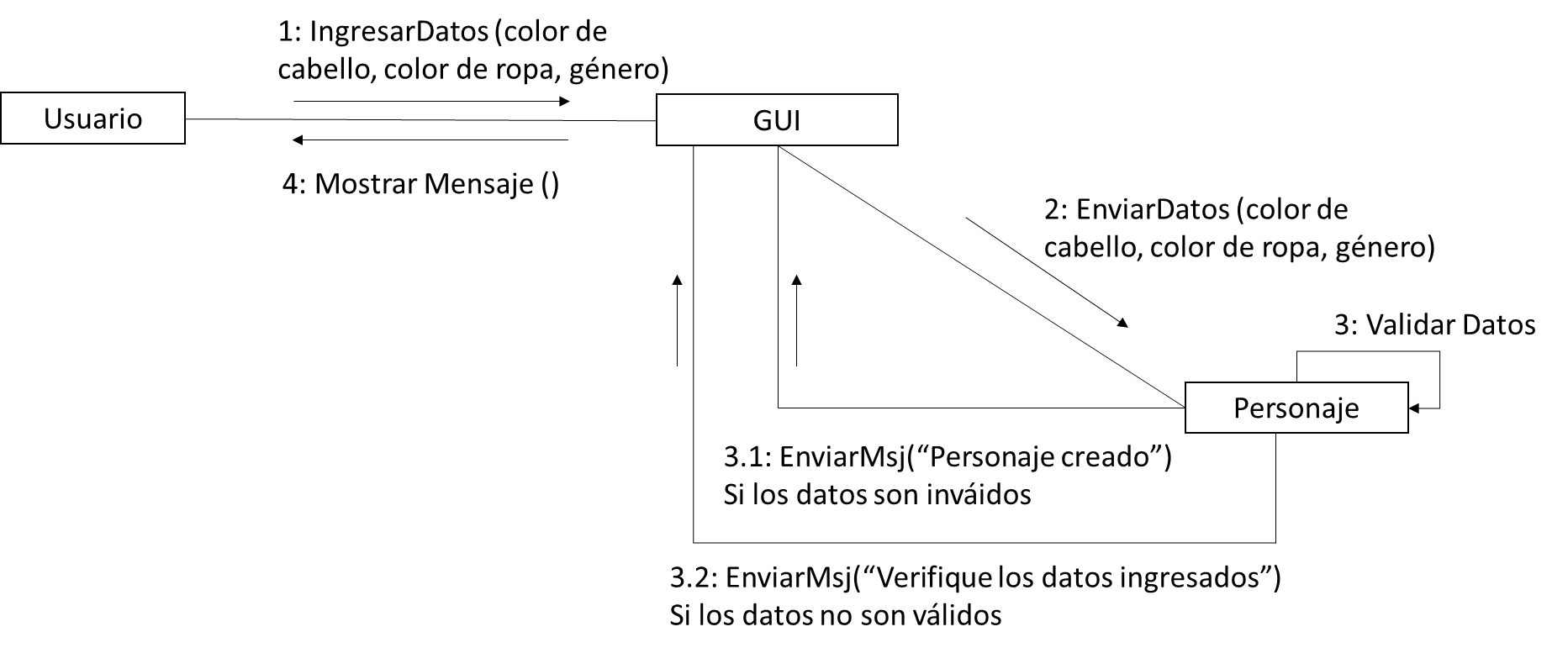
******

***Inicio de sesión.***

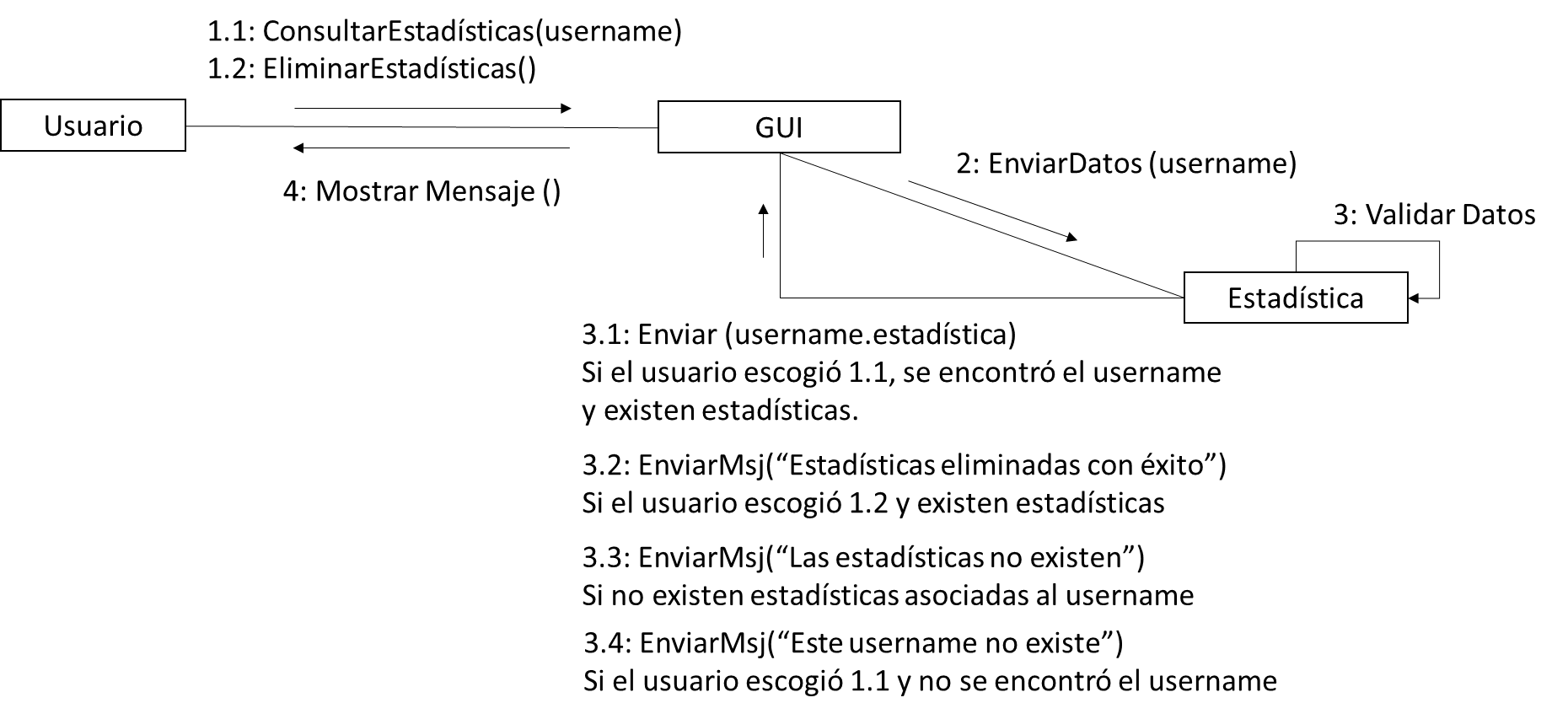
***Iniciar y unirse a partida.***

******

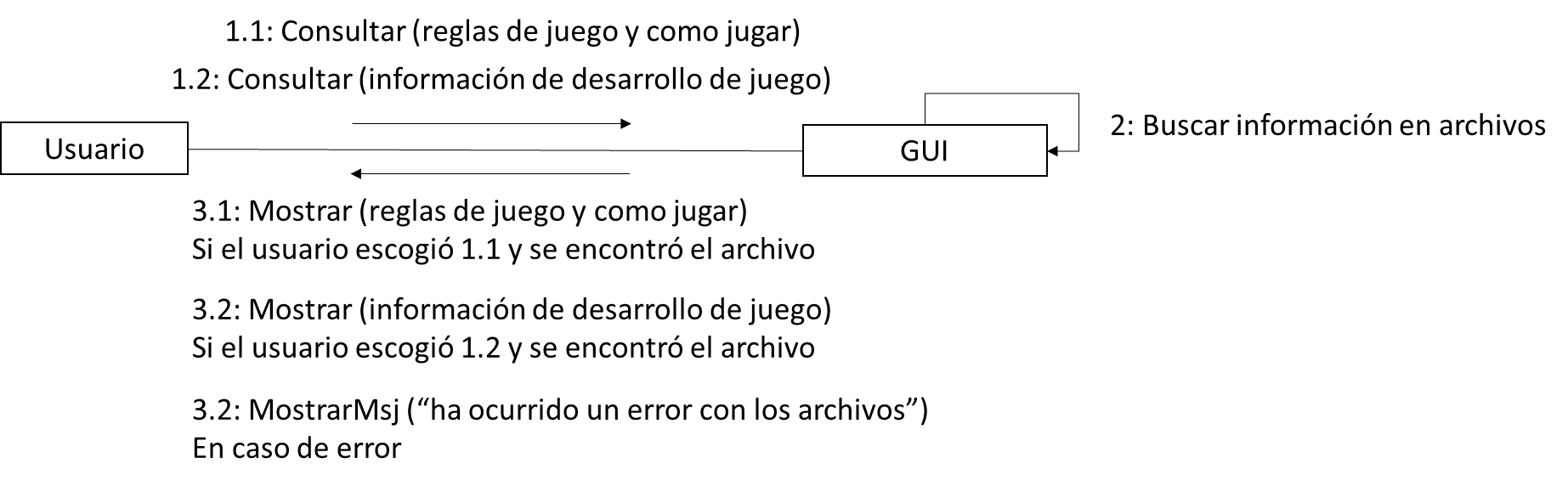
***Administrar personaje.***

****

**Administrar estadísticas.**

****

**Consultar información y ayuda.**

****

## **Diseño de Interfaz**

**Pantalla de inicio.** Es la primera parte de la interfaz con la que se encuentra el usuario, en donde puede elegir entre jugar una partida en modo de un jugador, ver estadísticas, iniciar sesión, registrarse, ver las instrucciones de juego e información de la aplicación



**Pantalla de crear o unirse a partida.** Se le ofrece al usuario la opción de disponer una partida o de unirse a una existente, así como también puede elegir el color de ropa de su personaje.



**Pantalla registrar usuario, iniciar sesión.** El usuario debe indicar las credenciales con las que se identificará a lo largo del uso del software.

**Pantalla de juego.** Pantalla que se despliega una vez se ha iniciado una partida, en donde se visualiza el desarrollo de un jugador dentro de uno de los 6 biomas preestablecidos, sus interacciones con los elementos que conforman a dicho bioma, así como también datos esenciales como la cantidad de vidas y la cantidad de esmeraldas obtenidas.

* Pradera:

¡¡Cosecha frutas de los árboles y carga comida a tu inventario!!



Consulta tu inventario cuando quieras, porta objetos de batalla y come alimentos para regenerar tus vidas.

* Bosque

¡¡Pelea contra los Mobs agresivos y gana esmeraldas!!

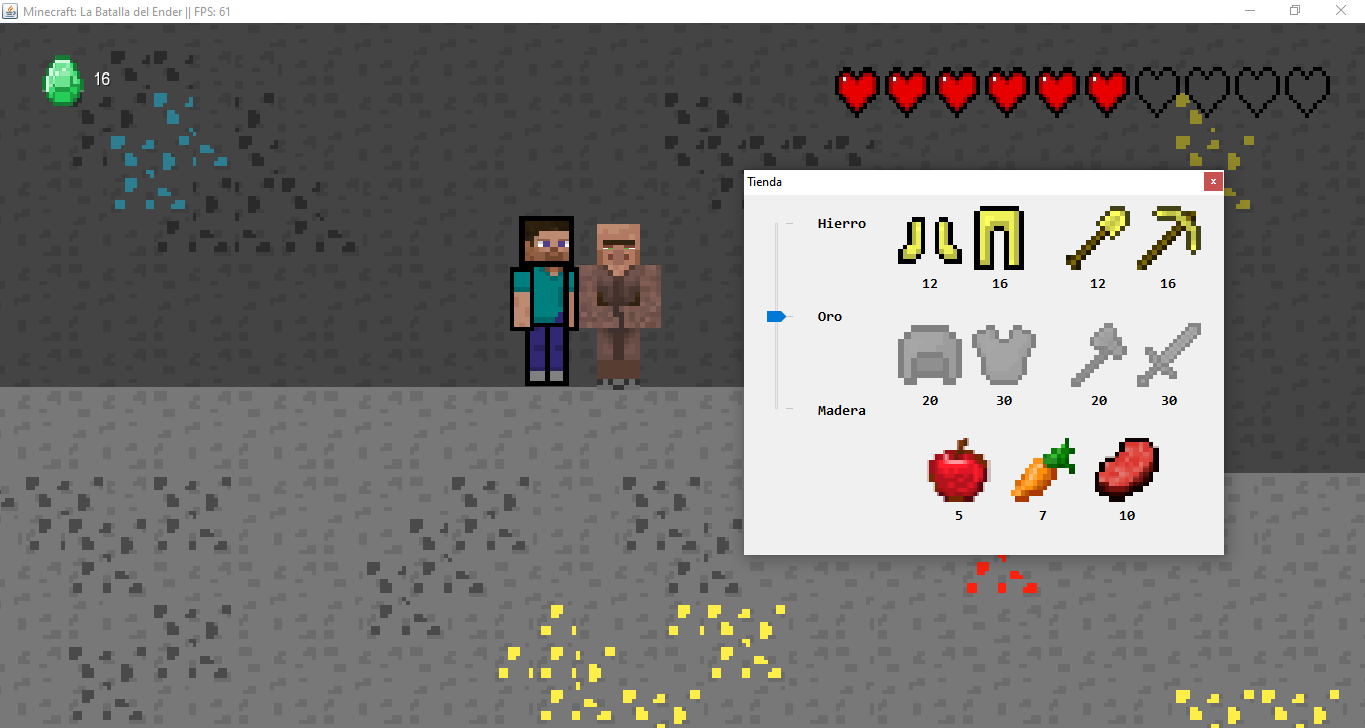
* Ender



* Infierno



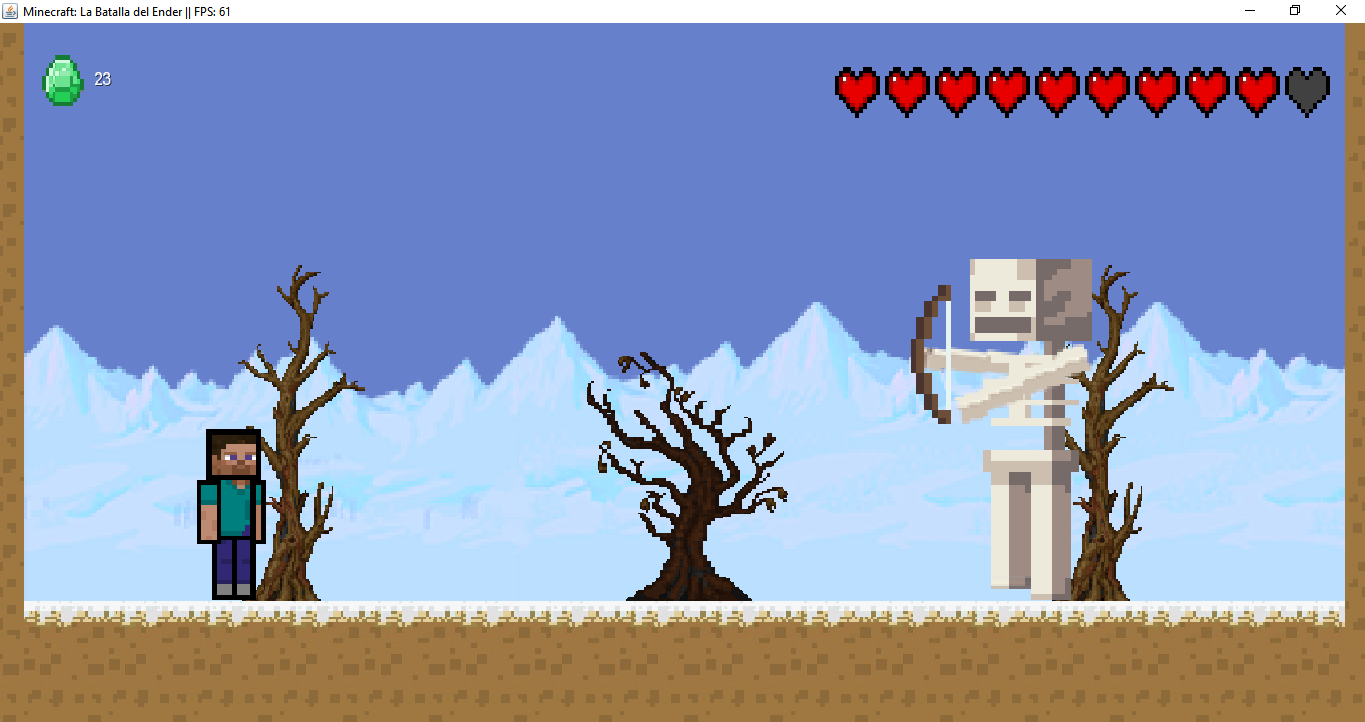
* Cueva

¡¡Visita la tienda del aldeano y compra comida y piezas de armadura para pelear contra los jefes de cada bioma!!

* Nieve Al matar a los Mobs pacíficos también obtienes comida y esmeraldas



Lucha contra los jefes que custodian cada bioma y gana una pieza de la armadura de diamante para tu inventario

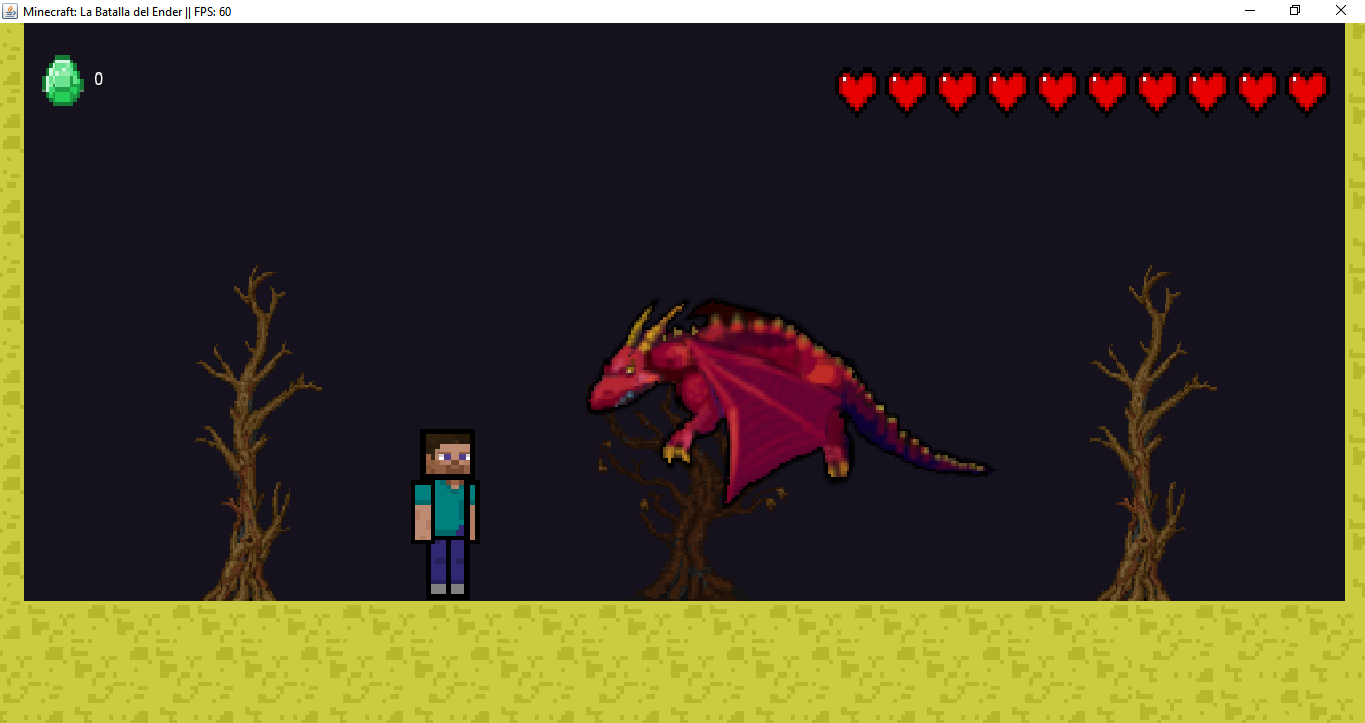


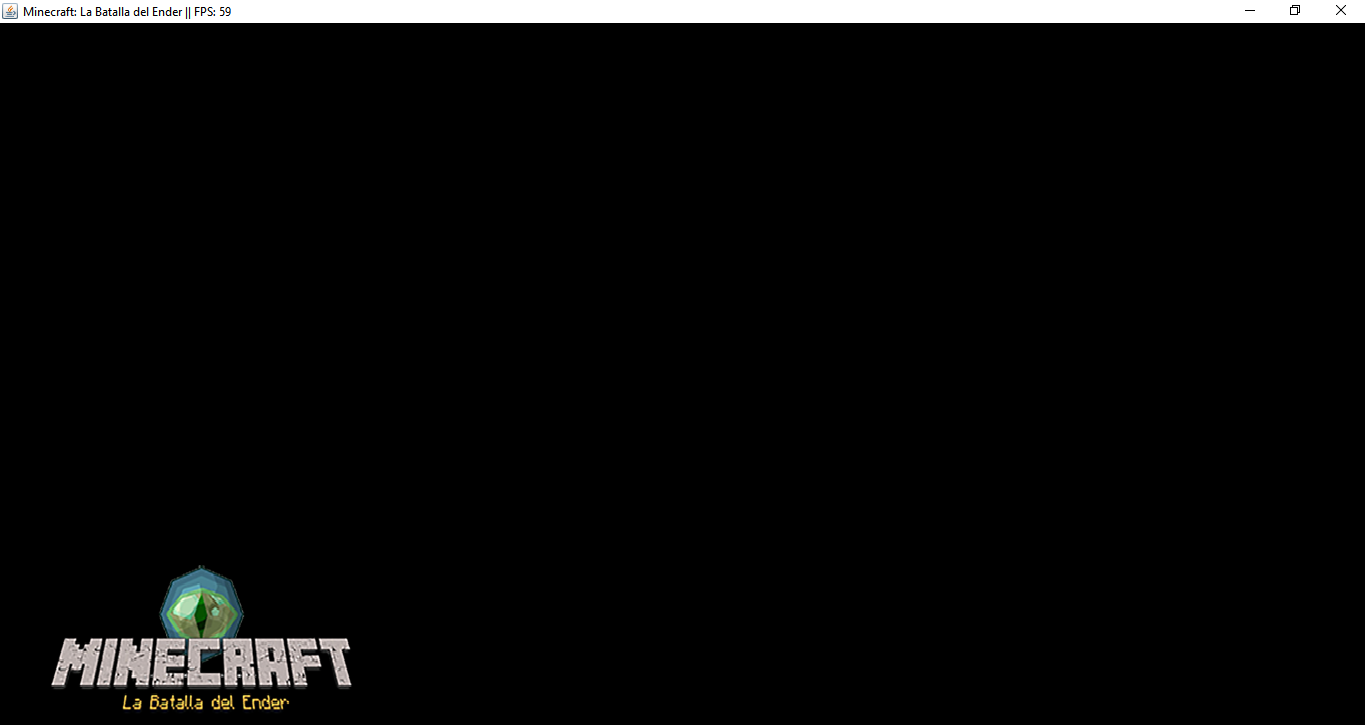


* Batalla Final (EnderDragon)

¿Ya obtuviste todas las piezas de la armadura de Diamante?

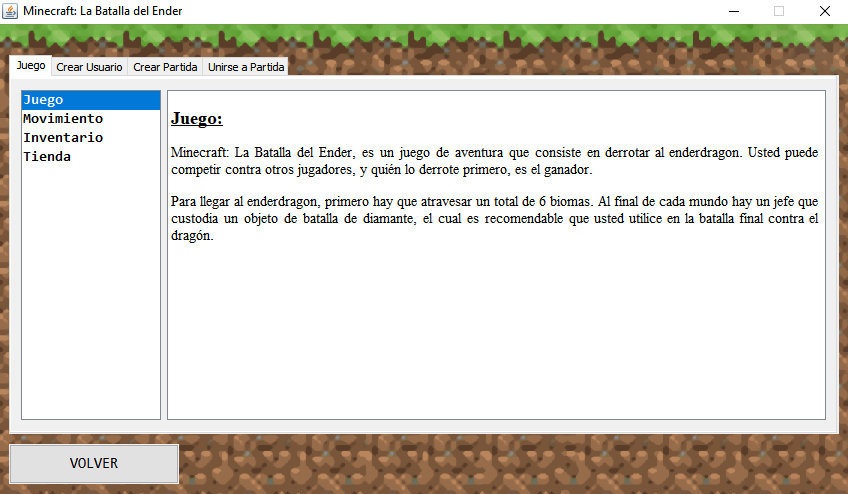
¡¡Derrota al EnderDragon para ganar!!



**Pantalla de carga.** Se muestra en el momento en el que un jugador se traslada de una habitación a otra (Bioma – Sala de combate) mientras el sistema carga los recursos necesarios para dibujarlos en pantalla.

**Pantalla de consulta de estadísticas.** El usuario puede ingresar el username de un jugador previamente registrado y ver las estadísticas obtenidas a lo largo de sus partidas jugadas.

**Pantalla Ayuda.** Contiene las instrucciones del juego acerca de como crear o unirse a una partida, registrarse en el juego y los controles para poder jugar.



**Pantalla Acerca de.** Contiene la información detallada acerca del desarrollo del software, las librerías externas utilizadas, nombre de los desarrolladores y versión actualizada.