**Análisis y Diseño de Software**

**Taller 4: Laberinto – Diseño**



**Realizado por:**

|  |  |  |
| --- | --- | --- |
| **Persona** | **Rol** | **Código Uniandes** |
| Carlos Ernesto González Vargas | Líder del Grupo | 200819123 |
| Sandra Milena Gómez Ríos | Líder de Planeación | 201110951 |
| Andrés Mauricio Erazo Benavides | Líder de Soporte | 201110949 |
| David Pérez Chibuque | Líder de Calidad | 201117818 |
| Willian Alejandro Idrobo Luna | Líder de Desarrollo | 201110544 |
| Erik Fernando Arcos Franco | Líder de Desarrollo | 201110856 |

**Índice de Contenido**

[**1.** **Introducción** 4](#_Toc307865953)

[**2.** **Objetivos** 4](#_Toc307865954)

[**3.** **Identificación y descripción de stakeholders** 4](#_Toc307865955)

[**4.** **Descripción Del Juego** 5](#_Toc307865956)

[**5.** **Reglas del Juego** 10](#_Toc307865957)

[**6.** **Requerimientos No Funcionales** 10](#_Toc307865958)

[**6.1.** **Restricciones** 10](#_Toc307865959)

[**6.2.** **Atributos de Calidad** 11](#_Toc307865960)

[**6.3.** **Árbol de Utilidad** 11](#_Toc307865961)

[**7.** **Definición de Actores** 12](#_Toc307865962)

[**8.** **Casos de Uso** 12](#_Toc307865963)

[**8.1.** **Casos de Uso del Jugador** 12](#_Toc307865964)

[**8.2.** **Casos de Uso de los Sitios Web Externos** 14](#_Toc307865965)

[**9.** **Diagrama de contexto** 14](#_Toc307865966)

[**10.** **Capacidades de un Jugador** 16](#_Toc307865967)

[**11.** **Definición del Laberinto** 17](#_Toc307865968)

[**12.** **Manejo del Área de Influencia** 18](#_Toc307865969)

[**13.** **Aspectos Críticos de Diseño** 21](#_Toc307865970)

[**14.** **Descomposición del Diseño** 22](#_Toc307865971)

[**15.** **Estructura de Datos** 29](#_Toc307865972)

[**16.** **Diagrama De Concurrencia** 34](#_Toc307865973)

[**17.** **Lecciones Aprendidas** 35](#_Toc307865974)

[**18.** **Conclusiones** 35](#_Toc307865975)

**Índice de Tablas**

[**Tabla 1. Stakeholders 4**](#_Toc307864425)

[**Tabla 2. Reglas del Juego 10**](#_Toc307864426)

[**Tabla 3. Árbol de Utilidad 11**](#_Toc307864427)

[**Tabla 4. Actores 12**](#_Toc307864428)

[**Tabla 5. Casos de Uso del Jugador 12**](#_Toc307864429)

[**Tabla 6. Casos de Uso de los Sitios Web Externos 14**](#_Toc307864430)

[**Tabla 7. Elementos del Juego 17**](#_Toc307864431)

[**Tabla 8. Selección del manejo del área de Influencia 18**](#_Toc307864432)

[**Tabla 9. Componentes nivel 1 23**](#_Toc307864433)

[**Tabla 10. Relaciones entre componentes sistema del juego laberinto 23**](#_Toc307864434)

[**Tabla 11. Componentes nivel 2 componentes cliente 25**](#_Toc307864435)

[**Tabla 12. Relaciones entre componentes sistema Cliente 25**](#_Toc307864436)

[**Tabla 12. Componentes nivel 2 componentes servidor 27**](#_Toc307864437)

[**Tabla 14. Relaciones entre componentes sistema Servidor 27**](#_Toc307864438)

[**Tabla 15. Estructura de Datos 29**](#_Toc307864439)

**Índice de Figuras**

[**Figura 1. Estados del jugador 5**](#_Toc307864440)

[**Figura 2. Actividades del Sistema 6**](#_Toc307864441)

[**Figura 3. Control del Turno 7**](#_Toc307864442)

[**Figura 4. Estados del Turno 8**](#_Toc307864443)

[**Figura 5. Ingreso al Juego 8**](#_Toc307864444)

[**Figura 6. Salir del Juego 9**](#_Toc307864445)

[**Figura 7. Casos de Uso del Jugador 12**](#_Toc307864446)

[**Figura 8. Casos de Uso de los Sitios Web Externos 14**](#_Toc307864447)

[**Figura 9. Diagrama de Contexto 15**](#_Toc307864448)

[**Figura 10. Atributos e inventario de un Jugador 16**](#_Toc307864449)

[**Figura 11. Jugadores en el Laberinto 17**](#_Toc307864450)

[**Figura 12. Visualización del Laberinto 18**](#_Toc307864451)

[**Figura 13. Definición de Zonas de Influencia y de Mapa 19**](#_Toc307864452)

[**Figura 14. Creación dinámica de zonas del Mapa 20**](#_Toc307864453)

[**Figura 15. Descomposición nivel 1 22**](#_Toc307864454)

[**Figura 16. Descomposición nivel 2 componente cliente 24**](#_Toc307864455)

[**Figura 17. Descomposición nivel 2 componente servidor 26**](#_Toc307864456)

[**Figura 18. Estructura de Datos 29**](#_Toc307864457)

[**Figura 19. Iniciar Aplicación 31**](#_Toc307864458)

[**Figura 20. Ingresar Jugador 32**](#_Toc307864459)

[**Figura 21. Controlar Turnos 33**](#_Toc307864460)

[**Figura 22. Diagrama de concurrencia 34**](#_Toc307864461)

**Análisis y Diseño de Software**

**Taller 4: Laberinto – Diseño**

1. **Introducción**

Como parte del proceso de aprendizaje de cómo realizar el análisis y diseño de un sistema informático se presenta el presente documento como una extensión del análisis del caso de estudio del laberinto, realizado en el taller 3 de la asignatura de Análisis y Diseño de Software de ECOS.

Este documento busca definir las decisiones de diseño tomadas para la implementación del sistema, para esto se presenta el análisis realizado en el taller 3 y se complementan ciertas secciones como por ejemplo el árbol de utilidad y los casos de uso del sistema. Con esta base se inicia el proceso de diseño para el taller 4.

1. **Objetivos**

* Analizar y entender el contexto en el que se desenvuelve el caso de estudio del laberinto.
* Realizar un modelo del mundo del aplicativo.
* Detallar las reglas de juego del aplicativo
* Definir el manejo de vecindades y zonas de influencia.
* Definir el manejo de la concurrencia del aplicativo.
* Realizar el diseño del laberinto Identificando los diferentes componentes que lo forman, como lo son fronteras, interfaces y servicios.

1. **Identificación y descripción de stakeholders**

Los Stakeholders del estudio son descritos a continuación:

Tabla 1. Stakeholders

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Stakeholder** | **Descripción** | **Responsabilidades** |
| **S1** | Arquitecto | Persona o grupo de personas que recibirán el análisis de la necesidad para diseñar la solución | Desarrollar el diseño arquitectural del sistema esperado |

**Análisis**

A continuación se realiza el análisis del juego del laberinto:

1. **Descripción Del Juego**

El sistema bajo análisis es la base para el inicio del diseño de un motor de soporte para una línea de juegos basados en el concepto de laberinto, el cual debe permitir a los usuarios jugar en red.

El juego inicia con un laberinto que puede ser recorrido por un conjunto de elementos nombrados elementos activos, que pueden estar asociados a un jugador o tener sus propios algoritmos de decisión. Adicionalmente, existen elementos pasivos distribuidos en el laberinto que modifican las características de los elementos activos cuando estos los consumen. En cada posición del laberinto pueden permanecer cualquier número de elementos pasivos y cualquier número de elementos activos que no sean agresivos entre sí, cuando dos elementos activos que son agresivos entre si se encuentran, se inicia una pelea que lleva a la salida de alguno de los elementos.



Figura 1. Estados del jugador

Cada jugador en el laberinto tiene un turno para realizar un movimiento valido, de acuerdo a la posición y el estado en el que se encuentre, el sistema realiza los turnos en paralelo y solo les da espera a los jugadores que afectan la jugada de otro jugador. El laberinto es infinito y se actualiza a medida que se realiza el juego, incluyendo elementos pasivos, elementos activos relacionados a jugadores y elementos activos con algoritmos propios.

Un jugador se encuentra en alguno de los siguientes estados de acuerdo a las acciones que se realicen durante el juego. Inicialmente el jugador para comenzar el juego debe **ingresar**, en donde se le asigna un elemento activo, unos elementos pasivos y se ubica dentro del laberinto en una posición, allí se le asigna un turno y se pone en estado de **espera** para ser incluido en el próximo turno.

El jugador permanece en el estado de **espera** hasta que se le asigna el turno de jugar para realizar su movimiento, allí el jugador puede pasar al estado en **movimiento** o **ataque** dependiendo de si se encuentra en una pelea o solo está recorriendo el laberinto.

En el estado en **movimiento** el jugador puede consumir elementos pasivos y recorrer el laberinto, moviéndose en su turno y regresando al estado en **espera**. El estado **ataque** ocurre cuando se llega a una posición donde existen más elementos activos que son agresivos con respecto a él, en este estado el jugador solo puede decidir **atacar**. Cuando se inicie el ataque este finaliza hasta que a alguno de los elementos en la pelea se le terminen su puntos de vida, al finalizar los puntos de vida el jugador pasa al estado **finalizado** para **salir** del juego. Adicionalmente desde el estado el movimiento el jugador puede decidir **salir** en cualquier momento y finalizar el juego.



Figura 2. Actividades del Sistema

El sistema comienza generando un laberinto para poder iniciar el juego, después de iniciado el juego se pueden empezar a adicionar los jugadores por medio de la acción de ingresar y en cualquier momento los jugadores pueden salir del juego. De manera paralela el sistema lleva control de los turnos asignando el turno para jugar de acuerdo a las dependencias entre jugadores y finalmente actualiza el laberinto con los nuevos ingresos, salidas y las actualizaciones que el sistema realiza sobre el laberinto.



Figura 3. Control del Turno

Para asignar el turno a un jugador el sistema verifica el estado del jugador y sus posibles movimientos válidos, una vez se da el turno al jugador se espera sus jugadas un tiempo T, si no se recibe respuesta se asume que no se realiza ningún movimiento, si el jugador se encontraba en estado de pelea el sistema evalúa si hubo un ganador para llamar la salida del perdedor y asignar los elementos pasivos del perdedor al ganador.



Figura 4. Estados del Turno

Cuando un jugador tiene un turno se valida el estado en el que esta y de acuerdo a este se tienen diferentes acciones, si el jugador se encuentra en medio de una pelea, solo puede atacar, mientras que si se encuentra en movimiento el jugador puede almacenar los elementos pasivos de la posición en la que se encuentra, consumir elementos pasivos almacenados y realizar el movimiento en este orden.



Figura 5. Ingreso al Juego

En cualquier momento se puede realizar el ingreso de un jugador, este se lleva a cabo estableciendo un elemento activo para el jugador, ya sea buscando entre los jugadores conocidos o ingresando uno nuevo, asignado su nombre, una vez se asigna el jugador activo, se le asignan los elementos pasivos, se sitúa el jugador en una posición del laberinto y se le asigna el siguiente turno de juego para ser incluido en el próximo ciclo del sistema.



Figura 6. Salir del Juego

Finalmente, en cualquier momento se puede realizarla salida de un jugador, esto incluye almacenar su estado, liberar los elementos pasivos que posea en ese momento dentro del laberinto en la posición actual y realizar las acciones para desconectarlo del sistema.

1. Reglas del Juego

A continuación se describen las reglas que definen el juego:

Tabla 2. Reglas del Juego

| **ID** | **Descripción** |
| --- | --- |
| **R001** | Un jugador siempre debe estar en unas coordenadas X, Y sobre un corredor. |
| **R002** | En cada punto del laberinto puede haber cero o más elementos, estos pueden ser del tipo (activo, pasivo). |
| **R003** | Elementos activos pueden ser agresivos o no agresivos entre ellos. |
| **R004** | En un mismo punto puede haber cualquier cantidad de elementos pasivos y activos, mientras no sean agresivos entre ellos. |
| **R005** | Los elementos pasivos pueden almacenarse para usarse más tarde. |
| **R006** | Un elemento de juego pude estar asociado con un jugador o ser independiente utilizar sus propios algoritmos para la toma de decisiones. |
| **R007** | Si dos elementos activos, agresivos entre sí, tratan de ocupar el mismo espacio, se produce una pelea. |
| **R008** | En la pelea siempre sale uno de los elementos cuando la vida de uno de los dos es menor o igual a cero. |
| **R009** | La pelea es iniciada por el atacante (El que ingresa al espacio) |
| **R010** | En cada iteración se le resta al nivel de vida del atacado (el nivel de ataque del atacante – nivel de defensa). |
| **R011** | Al final de la pelea el ganador obtiene los puntos de ataque y defensa del perdedor. |
| **R012** | Si en un punto el elemento se encuentra más un elemento agresivo respecto a él, pelea con cada uno de ellos en orden aleatorio, pero cada uno de ellos utiliza en cada pelea, la suma de niveles de defensa de los elementos atacados. |
| **R013** | Cada elemento activo se puede mover una posición en cualquier dirección válida en cada turno. |
| **R014** | En un turno el jugador tiene derecho a consumir cualquier número de elementos pasivos y hace un único movimiento válido. |
| **R015** | Deben poder aparecen elementos pasivos y activos durante la ejecución del juego |

1. **Requerimientos No Funcionales**
   1. **Restricciones**

* En el juego participaran miles de jugadores de manera simultánea
* Los turnos se deben realizar en paralelo respetando el orden pero aprovechando los casos en los que las jugadas de una persona son independientes a las jugadas de otra.
* Se le asigna a cada jugador un tiempo T para enviar la jugada, sino ha jugado en ese lapso se asume que decidió no moverse.
* Solo los jugadores que se encuentran en la zona de influencia de otro jugador deben esperar resultado de otras jugadas.
* La visualización de cada jugador es de 20 x 20 de su posición.
* La memoria de un jugador tiene un límite de 100 turnos.
  1. **Atributos de Calidad**

Para el sistema del laberinto se han identificado los siguientes atributos de calidad

* **Desempeño**: El laberinto es un escenario en el cual podrían estar miles de jugadores, por lo cual se hace necesario que la latencia del juego sea muy baja favoreciendo un sistema de turnos en paralelo que permita que varios jugadores in relación entre sí operen simultáneamente.
* **Modificabilidad**: El sistema debe ser lo suficientemente flexible para permitir la operatividad con diferentes plataformas, o la inclusión de nuevos sistemas. Igualmente debe considerarse que el sistema puede evolucionar a uno con mayor cantidad de usuarios o de mayor complejidad, lo cual podría derivar en el despliegue del mismo en diferentes ambientes.
* **Disponibilidad**: El sistema de consultas debe estar altamente disponible para permitir que los sitios externos puedan consultar información en cualquier momento.
  1. **Árbol de Utilidad**

Tabla 3. Árbol de Utilidad

| **Atributo** | **Medida** | **Descripción** | **Prioridad Negocio** | **Prioridad Arquitecto** |
| --- | --- | --- | --- | --- |
| **Desempeño** | **Latencia** | El sistema debe reducir al máximo la espera de turnos de los jugadores por medio del uso de turnos paralelos | Alta | Alta |
| **Modificabilidad** | **Escalabilidad** | El sistema debe ser escalable de manera que pueda soportar el despliegue en diferentes servidores con el fin de soportar altos volúmenes de usuarios conectados simultáneamente. | Normal | Alta |
| **Flexibilidad** | El sistema debe permitir la inclusión de nuevos sistemas. | Normal | Normal |
| **Portabilidad** | El sistema debe soportar las plataformas más populares en el momento y operar desde diferentes tipos de clientes | Normal | Baja |
| **Disponibilidad** | **Tiempo de Disponibilidad** | El sistema debe estar disponible para la consulta de estadísticas por parte de los sitios web externos | Baja | Baja |

1. **Definición de Actores**

Para el caso de estudio del laberinto tenemos se identificaron los siguientes actores

Tabla 4. Actores

|  |  |  |
| --- | --- | --- |
| **ID** | **Actor** | **Descripción** |
| **A1** | Jugador | Es el usuario que se conecta la sistema para jugar en el laberinto, el cual tiene asociado un nombre de usuario y un elemento activo con el cual juega. |
| **A2** | Sitios web externos | Sitios web que no tienen ninguna relación con el juego pero que pueden consultar la información sobre el estado actual del juego y sus jugadores |

1. **Casos de Uso**
   1. **Casos de Uso del Jugador**



Figura 7. Casos de Uso del Jugador

Tabla 5. Casos de Uso del Jugador

| **ID** | **Caso de Uso** | **Entradas** | **Precondiciones** | **Resultados** |
| --- | --- | --- | --- | --- |
| **CU01** | Crear Usuario | * Nombre Usuario * Login * Password | * El nombre de usuario es válido * El usuario no existe previamente en el sistema * La contraseña cumple con las políticas de seguridad | * Un nuevo usuario es creado en el sistema con el password seleccionado |
| **CU02** | Conectarse al laberinto | * Login del usuario * Password del usuario | * El usuario existe en el sistema * La clave es correcta | * El usuario ingresa al juego y es ubicado en el laberinto en una posición aleatoria y le son asignados 100 puntos de vida |
| **CU03** | Regresar al laberinto | * Login del usuario * Password del usuario | * El usuario existe en el sistema * La clave es correcta | * El usuario ingresa al juego y es ubicado en el laberinto en la posición que tenía antes de salir del juego |
| **CU04** | Realizar movimiento | * Dirección hacia la cual se desea mover el elemento | * El jugador se encuentra en su turno * El jugador debe tener un nivel de vida mayor a cero * La dirección debe ser valida | * El elemento que representa al jugador se mueve una posición hacia a la dirección deseada. * Si la posición está ocupada por un elemento activo agresivo el jugador entra a pelear |
| **CU05** | Tomar elemento pasivo |  | * El jugador no ha realizado ningún movimiento en el turno aun. * Existe un elemento pasivo en la posición en la que se encuentra el elemento | * El elemento pasivo es agregado al inventario del jugador |
| **CU06** | Consumir elemento pasivo | * Elemento pasivo a ser consumido | * El jugador no ha realizado ningún movimiento en el turno aun. * Existe al menos un elemento pasivo en el inventario del jugador | * Se aplican los efectos, temporales o permanentes, del elemento pasivo al elemento activo del jugador * El elemento pasivo usado es retirado del inventario |
| **CU07** | Pelear | * Nivel de vida * Nivel de ataque * Nivel de defensa | * La posición esta previamente ocupada por uno o más elementos agresivos al jugador | * Ganador: El jugador obtiene los puntos de el o los defensores de la posición * Perdedor: El jugador es retirado del laberinto. |
| **CU08** | Salir del laberinto |  |  | * La información del elemento activo del jugador es almacenada. * Los elementos pasivos del jugador son descartados en la posición en la que se encuentra el elemento * El elemento es retirado del laberinto * El jugador es desconectado del juego. |
| **CU11** | Chatear con otro jugador | * Id del jugador | * El jugador debe estar conectado * El jugador a contactar debe estar en la misma área de influencia del jugador | * Se establece una conexión entre ambos jugadores y se despliega una interfaz en la cual los jugadores pueden intercambiar mensajes |

* 1. **Casos de Uso de los Sitios Web Externos**



Figura 8. Casos de Uso de los Sitios Web Externos

Tabla 6. Casos de Uso de los Sitios Web Externos

| **ID** | **Caso de Uso** | **Entradas** | **Precondiciones** | **Resultados** |
| --- | --- | --- | --- | --- |
| **CU09** | Consultar estado actual del juego |  | * Existe un juego en el sistema | * Información sobre el estado actual del juego. |
| **CU10** | Consultar mejores jugadores |  | * Existe un juego en el sistema | * Listado con los mejores jugadores en el histórico del juego y sus estadísticas (puntajes, batallas, etc.). |

1. Diagrama de contexto



Figura 9. Diagrama de Contexto

El motor a diseñar debe contar con la posibilidad de soportar jugadores en línea. Estos usuarios pueden acceder al juego a través de un la cuenta que se tenga configurada en el sistema del juego, la cual es verificada contra el sistema LDAP. Luego de esta verificación el jugador comienza a participar en el laberinto a medida que le corresponda su turno. A medida que se desarrolla el juego, este genera diferentes datos los cuales sirven para ser consultados y llevar estadísticas de los jugadores y el estado del juego.

**Diseño**

Basado en el análisis anterior se realiza el diseño del juego del laberinto que se presenta a continuación:

1. **Capacidades de un Jugador**

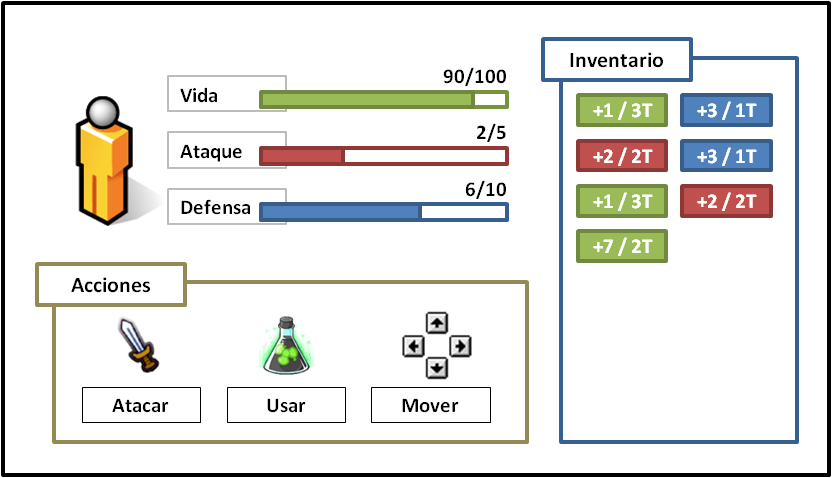


Figura 10. Atributos e inventario de un Jugador

La imagen muestra las capacidades generales que describen a un jugador dentro del laberinto así como las acciones que puede realizar, en el diagrama anterior se tiene que:

* Un jugador se define por 3 características diferentes: Vida, Ataque y Defensa
* Un jugador inicia con un valor de 100 de vida, 1 de ataque y 1 de defensa, estos valores se pueden alterar por medio de elementos pasivos.
* Un jugador puede realizar diferentes acciones durante su turno: Atacar, Usar un ítem del inventario o moverse
* Un ítem, o elemento pasivo puede ser almacenado por el usuario para ser usado posteriormente y se puede ver en su inventario.
* Los elemento pasivo puede alterar cada una de las características de un jugador: Vida, Ataque y Defensa, definidas en el diagrama por los colores Verde, Rojo, Azul respectivamente, de esta manera un elemento pasivo altera cada una de estas características durante una determinada cantidad de turnos.

1. **Definición del Laberinto**

El funcionamiento de un laberinto se puede definir por medio de la siguiente imagen:

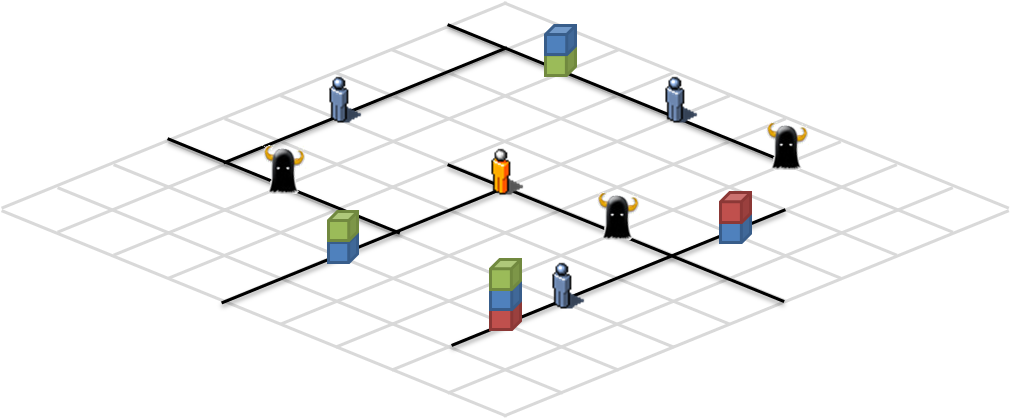


Figura 11. Jugadores en el Laberinto

Donde cada uno de los elementos representa lo siguiente:

Tabla 7. Elementos del Juego

|  |  |
| --- | --- |
| **Elemento** | **Definición** |
|  | Elemento pasivo de modificación de Vida, puede haber varios elementos pasivos en una misma posición. |
|  | Elemento pasivo de modificación de Defensa, puede haber varios elementos pasivos en una misma posición. |
|  | Elemento pasivo de modificación de Ataque, puede haber varios elementos pasivos en una misma posición. |
|  | Elemento agresivo controlado por el sistema |
|  | Jugador |
|  | Elemento agresivo controlado por otro jugador |
|  | Corredor dentro del laberinto |

Aunque el sistema conoce la distribución de todos los elementos, el jugador no conoce el estado actual del laberinto, de hecho solo conoce los caminos por los que ha recorrido, y su visualización es de 20x20 cuadros a su alrededor y solo recuerda sus últimos 100 turnos, de esta manera, para un jugador la visualización del laberinto en un momento dado podría ser la siguiente:

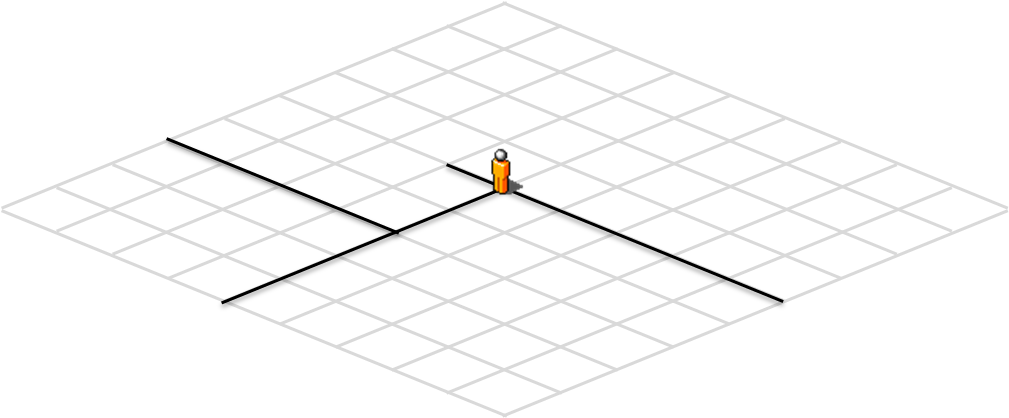


Figura 12. Visualización del Laberinto

1. **Manejo del Área de Influencia**

Para el manejo del área de influencia dentro de un laberinto se analizaron 3 opciones diferentes propuestas por los integrantes del grupo, cada una de estas opciones fue expuesta y se votó por cada una de ellas para seleccionar la opción que será desarrollada, esta información se puede ver en la siguiente tabla.

Tabla 8. Selección del manejo del área de Influencia

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nombre** | **Descripción** | **Pros** | **Contras** | **Votos** |
| **Corredor** | El área de influencia se asocia a un corredor y sus intersecciones. | Un corredor puede contener una o varias zonas de influencia dependiendo de su longitud. | Se debe prestar especial cuidado a las intersecciones. | 3 |
| **Zonas** | Un área cuadrada representa la zona de influencia y controla todos los corredores que la cruzan. | Se disminuye la cantidad de procesos que corren en el servidor en relación a la zona de influencia. | Si la zona es demasiado grande, el desempeño de un jugador se puede ver afectado al tener que esperar el procesamiento de muchos jugadores. | 1 |
| **Jugador** | Cada uno de los jugadores tiene su propia zona de influencia. | Cada jugador controla sus propias vecindades | La cantidad de procesos es igual a la cantidad de jugadores, lo cual puede representar sobrecarga, además de que se debe idear una manera de comunicar estas diferentes áreas. | 2 |

A continuación se describe la opción seleccionada tal como lo muestra la siguiente figura, en la cual se reducen las dimensiones a fin de poder describir mejor el concepto:

* Las líneas negras representan los corredores, un corredor tiene un punto de inicio, una dirección y una longitud opcional, la cual, si no está presente indica que el corredor es infinito.
* Los recuadros azules representan las zonas de influencia, las cuales están asociadas a un corredor, pueden existir una o más zonas de influencia por corredor tal como se explicará más adelante.
* Las zonas rojas indican zonas de influencia de las intersecciones.
* El recuadro verde alrededor de mapa indica una zona de frontera empleada para generar dinámicamente nuevas zonas del mapa.



Figura 13. Definición de Zonas de Influencia y de Mapa

En la figura anterior se debe resaltar lo siguiente:

* **Zona de Mapa:** Una zona de un mapa es un cuadrado de área X por X, el cual es generado al momento de que se carga el sistema, a medida que los jugadores se mueven dentro de esta zona pueden llegar a la zona verde, la zona de frontera, en este momento el sistema crea una nueva zona del mapa antes de que el jugador llegue al borde del mismo, esto permite no cargar todo el mapa desde el inicio a la vez que se crean nuevas zonas de manera dinámica a medida que sea necesario. Esto se puede observar en la imagen de la página siguiente.
* **Zona de Influencia:** La zona azul, es la zona de influencia, dentro de esta zona se debe determinar el orden en el cual los jugadores pueden moverse, esta zona mide 100 posiciones, y es centralizada en el servidor, de manera que si hay un jugador en la posición 10 y otro en la 85, pueden jugar paralelamente, pero si se encuentran a menos de 4 cuadros de distancia, el sistema asigna turnos. Si un corredor mide 100 o menos unidades, puede ser controlado por una sola zona de influencia, pero si es más largo debe dividirse en varias zonas tal como lo muestra in corredor inferior de la anterior imagen. Dos zonas continuas se unen por medio de una zona de intersección al igual que dos corredores que se cruzan, esto se explica a continuación.
* **Zona de Intersección:** La zona roja o zona de intersección sirve de punto de unión entre las diferentes zonas de influencia. Dado que al final de una zona de influencia puede continuar otra, o incluso superponerse con otra, es necesario dar un manejo especial a estas situaciones, las zonas de intersección manejan la complejidad de varios corredores convergentes y miden 2 unidades desde el punto de intersección.

 .



Figura 14. Creación dinámica de zonas del Mapa

La imagen anterior muestra 3 zonas del mapa, inicialmente solo existía la esquina inferior izquierda, pero a medida que se mueven los jugadores se crearon las otras dos zonas si ellos se mueven en dirección a la frontera.

1. **Aspectos Críticos de Diseño**

Como se definió en nuestro árbol de utilidad el principal atributo de calidad que guía nuestro diseño es el desempeño seguido de la modificabilidad.

Para iniciar se definió utilizar la arquitectura cliente-servidor para separar responsabilidades, el cliente manejara principalmente la visualización, el servidor manejara la conexión con los clientes y procesamiento de jugadas. Con el fin de favorecer desempeño en el servidor y este no se sobrecargue de responsabilidades. A parte de esto se decidió tomar como estrategia Eager Acquisition en la carga del mapa de forma local y realizar la visualización en el cliente luego del procesamiento de los movimientos de los jugadores realizado por el servidor. De esta forma no se tiene que cargar la información del mapa en cada movimiento sino que este se actualice cada vez que el jugador cambie de mapa.

Otras características adicionales que ofrece el sistema es que la comunicación entre el cliente y servidor se realiza por medio de sockets, utilizando una estructura de datos basada en bits para asegurar una velocidad alta. También se empleó una comunicación alternativa para la autenticación por medio de servicios web para posteriormente abrir el canal de comunicación directa entre el cliente y servidor.

En cada sistema (Servidor, Cliente) se dividieron los componentes con responsabilidades claras, tener una alta cohesió, bajo acoplamiento de esta forma favorecer la modificabilidad del sistema. Estos sistemas se vieron altamente influenciados por los motivadores de concurrencia y modificabilidad, lo cual se refleja en los componentes que tienen responsabilidades de manejar las zonas de influencia, zona de mapas, Master, Chat, Seguridad, Persistencia, Core, etc.

1. **Descomposición del Diseño**

Para el diseño inicial se definieron dos componentes principales, primero un cliente presente en cada equipo de un jugador, encargado de recibir las jugadas y presentar la interfaz grafica al jugador, segundo un servidor encargado de administrar el core del juego del laberinto y mediar la comunicación entre los diferentes clientes.

Adicionalmente aparecieron componentes adicionales para satisfacer los requerimientos relacionados a la seguridad y la consulta de la información del juego por parte de sistemas externos.



Figura 15. Descomposición nivel 1

A continuación se encuentra la descripción de cada uno de los componentes del diagrama:

Tabla 9. Componentes nivel 1

| **Componente** | **Descripción** |
| --- | --- |
| **Cliente laberinto** | El componente cliente se encarga de presentar la interfaz de grafica al jugador y enviar los mensajes al servidor, donde se procesan para realizar las acciones del juego.  Dentro del cliente se encuentra una copia del mapa y se actualiza en cada jugada los elementos activos y pasivos que se encuentran en su vencida de posiciones.  Los casos de uso relacionados a este componente son: del CU001 al CU008 |
| **Servidor Laberinto** | El componente servidor es el encargado de manejar toda la lógica del juego entre los diferentes clientes, proporcionándoles un modelo de laberinto donde jugar y coordinando los turnos de juego, además de almacenar toda la información de los jugadores y de las estadísticas de juego  Los casos de uso relacionados a este componente son: CU009, CU010 |
| **LPDA <database>** | El componente con la información de usuarios que pueden acceder al laberinto.  Los casos de uso relacionados a este componente son: CU001, CU002, CU008 |
| **Estadísticas laberinto <database>** | El componente con la información de las estadísticas del laberinto.  Los casos de uso relacionados a este componente son: CU009, CU010 |
| **Sistema externos** | Sistemas desconocidos que van a consultar la información de las estadísticas del juego.  Los casos de uso relacionados a este componente son: CU009, CU010 |

Para tener un mejor entendimiento del diagrama, especificamos a continuación la relación que hay entre los componentes del sistema.

Tabla 10. Relaciones entre componentes sistema del juego laberinto

| **Componentes** | **Interacción** |
| --- | --- |
| **ServidorLaberinto – ClienteLaberinto**  **(ISeguridad)** | El servidor provee servicios para que el componente cliente laberinto pueda realizar la autenticación del usuario y reciba como respuesta los datos de conexión de juego con el servidor. |
| **ServidorLaberinto – ClienteLaberinto**  **(sockets)** | Es el canal de comunicación utilizado para enviar los datos del juego, entre los clientes (jugadores) y el servidor, por este medio se transmiten los movimientos de los clientes y el chat de las diferentes zonas de influencia. |
| **LDAP - ServidorLaberinto** | Servidorlaberinto consulta la información del usuario en LDAP y verifica si los datos son correctos. |
| **EstadisticasLaberinto - ServidorLaberinto** | Servidorlaberinto almacena la información del juego y las estadísticas generadas del mismo, para que se pueda consultar posteriormente por sistemas externos. |
| **SistemasExternos – EstadisticasLaberinto** | EstadísticasLaberinto expone un servicio para que los sistemas externos puedan acceder a la información histórica del juego. |

Se considero necesario descomponer con un detalle mayor los elementos del diseño para tener las responsabilidades más claras en cada componente, en el siguiente diagrama se aparecen los componentes que soportan y proveen funcionalidad en el componente cliente.



Figura 16. Descomposición nivel 2 componente cliente

El sistema del juego de laberinto en el modulo del cliente, tiene la responsabilidad comunicarse y enviar la información al servidor, además se encarga controlar la visualización y configuración inicial para el inicio del juego.

A continuación se encuentra la descripción de cada uno de los componentes del diagrama:

Tabla 11. Componentes nivel 2 componentes cliente

| **Componente** | **Descripción** |
| --- | --- |
| **Core** | Coordina la interacción entre los componentes y es el encargado de la comunicación de información hacia el servidor.  Los casos de uso relacionados a este componente son: 01,02,0304,05,06,07, 08,09 |
| **Seguridad** | Controla la lógica para la autenticación del usuario.  Los casos de uso relacionados a este componente son: 02 |
| **Comunicación** | Abre el canal de comunicación entre cliente y el servidor, provee los métodos para la comunicación entre ellos.  Los casos de uso relacionados a este componente son: 04,05,06,07 |
| **Starter** | Contiene la información de configuración para el inicio del juego, se encarga de iniciar el aplicativo y preparar los componentes para que se pueda realizar el juego. |
| **Cache Mapa** | Almacena una copia del mapa en el que se encuentra el jugador actualmente. Esto no incluye los elementos activos ni pasivos, solo los corredores del laberinto y las posiciones en donde ha estado el jugador.  Los casos de uso relacionados a este componente son:02,04 |
| **Visualización** | Maneja la lógica de presentación y visualización del juego.  Los casos de uso relacionados a este componente son: 01,02,03,04,05,06,07,08 |
| **Chat** | Se encarga de manejar la lógica de chat en el cliente. Maneja la lógica del patrón suscriptor para la comunicación con el servidor y clientes.  Los casos de uso relacionados a este componente son: CU011 |

Para tener un mejor entendimiento del diagrama, especificamos a continuación la relación que hay entre los componentes del sistema.

Tabla 12. Relaciones entre componentes sistema Cliente

| **Componentes** | **Interacción** |
| --- | --- |
| **Seguridad - Servidor** | El servidor provee servicios para que el componente Seguridad pueda realizar la autenticación del usuario y otros servicios de seguridad que requiera. |
| **Core - Seguridad** | Seguridad expone una interfaz para que el Core verifique si la autenticación del usuario fue exitosa y se pasan los datos de configuración de esa forma permitir el inicio de la comunicación con el servidor. |
| **Core - Comunicacion** | Comunicación expone los servicios para la creación y procesamiento de los mensajes. |
| **Core -Starter** | Starter es llamado externamente por los usuarios y tiene los datos de inicio Core iniciar. |
| **Core - CacheMapa** | CacheMapa recibe la información del mapa por medio del Core para almacenar una copia local y esa información se cargue cada vez que se realice un cambio de mapa. |
| **Core - Visualizacion** | Visualizacion obtine la información del juego que el usuario envía y es enviada al Core para comunicarla al servidor. Además Core envía a Visualización el estado del mapa y de los demás elementos para ser visualizada en el cliente. |
| **Core – Chat** | Chat expone los servicios para el procesamiento de los mensajes que se reciben y envían en el Core. |
| **Core - Servidor** | Core mantiene la comunicación hacia el servidor, por medio de este pasan todos los mensajes del cliente al servidor y viceversa. Se encarga de coordinar los demás componentes del cliente para administrar la lógica del juego en el cliente. |

En este diagrama se describe los componentes que soportan y proveen funcionalidad del componente servidor.



Figura 17. Descomposición nivel 2 componente servidor

El sistema del juego de laberinto en el modulo del servidor, tiene la responsabilidad de proveer a los clientes la conexión hacia el juego, controla la información y la lógica del juego y se encarga de almacenar y consultar la información de los usuarios y el estado del juego.

Tabla 13. Componentes nivel 2 componentes servidor

| **Componente** | **Descripción** |
| --- | --- |
| **Persistencia** | Consulta y almacena la información relacionada del juego y los usuarios |
| **Seguridad** | Controla la lógica para la autenticación del usuario.  Los casos de uso relacionados a este componente son: 01,02,03 |
| **Comunicación** | Abre el canal de comunicación entre servidor y los clientes, provee los métodos para la comunicación entre ellos.  Los casos de uso relacionados a este componente son: 04,05,06,07,08 |
| **Master** | Coordina la interacción entre los componentes encargados de manejar la lógica del mapa y centraliza las comunicaciones entre los clientes y el servidor.  Los casos de uso relacionados a este componente son: 01,02,03,04,05,06,07,08 |
| **Controlador mapa** | Controla la creación de elementos activos, pasivos y jugadores en el mapa, es el primer control para la realización del movimiento de los jugadores. |
| **Zona mapa** | Se encarga de controlar la generación del mapa y de manejar los movimientos de los jugadores entre zona de influencia |
| **Elementos Activos** | Son los que representan los elementos en el juego para los jugadores y los bots que tienen inteligencia para actuar como elementos agresivos. |
| **Elementos Pasivos** | Representan elementos que modifican los atributos de los elementos activos entre el laberinto. |
| **Zona influencia** | Controla la lógica de juego, movimientos, ataques, turnos en la zona de influencia. |
| **Chat** | Se encarga de manejar la lógica de chat entre el servidor y los clientes. |

Para tener un mejor entendimiento del diagrama, especificamos a continuación la relación que hay entre los componentes del sistema.

Tabla 14. Relaciones entre componentes sistema Servidor

| **Componentes** | **Interacción** |
| --- | --- |
| **Cliente - Seguridad** | Seguridad expone un servicio para que el Cliente pueda realizar la autenticación y obtenga los datos de conexión para el juego. |
| **LDAP - Seguridad** | Seguridad consulta la información del usuario en LDAP y verifica si los datos son correctos. |
| **Seguridad - Master** | Master inicia la conexión con el cliente luego de la autenticación del usuario por parte de Seguridad. |
| **Persistencia - Master** | Consulta los datos de los jugadores. |
| **Persistencia - Controlador Mapa** | ControladorMapa envía los datos de los jugadores y el mapa para las estadísticas del juego. |
| **Persistencia - EstadisticasJuego** | Persistencia envía los datos del juego al componente EstadisticasJuego para que esté disponible a los sistemas externos. |
| **Master - Comunicacion** | Comunicación expone los servicios para la creación y procesamiento de los mensajes. |
| **Controlador Mapa - Master** | ControladorMapa envía información del estado del mapa y de los usuarios a Master. Por otro lado Master asigna a ControladorMapa los usuarios que se encuentran en el mapa para que sean creados e inicien el juego. Además se comunican cada vez que un usuario se desconecta o se va salir del mapa. |
| **Controlador Mapa - Zona Mapa** | Zona Mapa comunica a ControladorMapa si un usuario cambia de mapa y es necesario asignarlo a otra zona, también recibe la información de los jugadores que son asignados a su zona. |
| **Zona Influencia - Cliente** | Zona Influenciaenvía información de los jugadores y mapa al Cliente para que sea procesada y visualizada en el mapa del cliente. El Cliente envía la información de los movimientos a Zona Influencia para que procese la jugada. |
| **Zona Mapa - Comunicacion** | ZonaMapa envía la información a Comunicación para que sea preparada y posteriormente enviada al cliente por sockets. |
| **Zona Mapa – Elementos Activos** | Zona Mapa emplea Elementos Activos para la creación de estos elementos en el mapa. Elementos Activos comparte información de su estado y atributos con Zona Mapa. |
| **Zona Mapa – Elementos Pasivos** | Zona Mapa emplea el componente ElementosPasivos para la creación de estos elementos en el mapa. |
| **Zona Mapa – Zona Influencia** | Zona Influencia permite la asignación de jugadores en su zona por parte de Zona Mapa. Zona Mapa permite recibir el estado de los jugadores y la zona de influencia en la que se encuentran para el control de estos. |
| **Zona Influencia - Chat** | Zona Influencia y Chat se usan para la comunicación de los jugadores. Chat prepara el mensaje de los jugadores y por medio de ZonaInfluencia es enviado a los jugadores que se encuentren en la misma zona de influencia. |

1. **Estructura de Datos**



Figura 18. Estructura de Datos

Tabla 15. Estructura de Datos

| **Entidad** | **Descripción** | **Atributos y asociaciones** |
| --- | --- | --- |
| **Laberinto** | Representa el laberinto | * **fechaFin:** Fecha final de existencia del laberinto * **fechaInicio:** Fecha en que se crea el laberinto * **numHistoricoMovimientos:** Numero de movimientos que se almacenaran por jugador * **tiempoTurno:** Tiempo que toma un turno de un jugador en el laberinto * **[1..\*] contiene:** Conjunto de áreas que contiene el laberinto |
| **Area** | Representa la áreas que contiene un laberinto | * **idArea:** * **alto:** * **ancho:** * **[1..\*] contiene:** Conjunto de zonas de influencia que contiene el área |
| **ZonaInfluencia** | Representa una zona de influencia del área | * **idZonaInfluencia:** identificador de la zona * **[1..\*] contiene:** Conjunto de corredores que contiene la zona de influencia * **[0..1] tiene:** información del chat |
| **Corredor** | Representa un corredor del laberinto | * **puntoInicio:** Es la coordenada de inicio del corredor * **puntoFin:** Es la coordenada de finalización del corredor * **[\*] contiene:** Conjunto de coordenadas que hacen parte de un corredor |
| **Coordenada** | Representa un punto en el laberinto el cual puede estar o no ocupado por un elemento | * **idCorredor:** identificador del corredor * **puntoInicio:** punto de inicio * **puntoFin:** punto final * **[0..\*] tiene:** Son los elementos, activos o pasivos que se encuentran en la coordenada del laberinto * **[0..1] tiene:** información de la pelea ejecutada en la coordenada |
| **Pelea** | Representa una pelea originada por dos o más jugadores | * **idPelea:** identificador de la pelea * **ganador:** Identifica el jugador ganador de la pelea * **[0..\*] tiene:** Jugadores involucrados en la pelea |
| **Elemento** | Representa un elemento, activo o pasivo que se encuentra en el laberinto | * **idElemento:** Especifica si la instancia del elemento es un elemento pasivo o no lo es * **símbolo:** Es el símbolo del elemento **[1] pertenece:** Es el laberinto al cual pertenece el elemento. |
| **Elemento Pasivo** | Representa un elemento pasivo del juego | * **modificadorPermanenteAtaque:** Modifica permanentemente el atributo de ataque del elemento activo que lo toma o usa. * **modificadorPermanenteDefensa:** Modifica permanentemente el atributo de defensa del elemento activo que lo toma o usa. * **modificadorPermanenteVida:** Modifica permanentemente el atributo de vida del elemento activo que lo toma o usa. * **modificadorTemporalAtaque:** Modifica temporalmente el atributo de ataque del elemento activo que lo toma o usa. * **modificadorTemporalDefensa:** Modifica temporalmente el atributo de defensa del elemento activo que lo toma o usa. * **modificadorTemporalVida:** Modifica temporalmente el atributo de vida del elemento activo que lo toma o usa. |
| **Elemento Activo** | Representa a un elemento activo del juego | * **nivelAataque:** Nivel de ataque del elemento activo * **nivelDefensa:** Nivel de defensa del elemento activo * **nivelVida:** Nivel de vida del elemento activo * **esAgresivo:** Nivel de vida del elemento activo * **[0..\*] tiene:** Representa los elementos pasivos que el elemento activo posee |
| **Jugador** | Representa a un elemento activo que es un jugador humano | * **nombre:** Es el nombre del jugador * **puntaje:** Es el puntaje del jugador * **turno:** Es el turno del jugador |
| **Robot** | Representa a un elemento activo que es independiente (no humano) | * **algoritmoMovimiento**: algoritmo que define el movimiento del robot |
| **HistoricoMovimiento** | Representa la memoria de los últimos movimientos de un jugador | * **fechaJugada:**  fecha de realizacion de la jusgadas * **numMovimientos:** número de movimientos que lleva el jugador * **[1..\*] tiene:** Coordenadas en las que ha estado el jugador |
| **Usuario** | Representa un usuario logueado en el juego | * **login:** login del usuario * **direccionIP:**  dirección ip del usuario * **puerto:**  puerto por el que se conecta un usuario * **servidorCredenciales:** servidor de credenciales donde se crea el usuario * **[0..1] tiene:** información del jugador |
| **Chat** | Representa el envío y recepción de los mensajes entre los jugadores de una misma zona de influencia | * **idChat:** identificador del chat * **fecha:** fecha de creación del chat * **[0..1] tiene:** información de los usuarios * **[0..1] tiene:** mensajes enviados |



Figura 19. Iniciar Aplicación

Al iniciar la aplicación de juego, se crea el laberinto y toda su información relaciona como las áreas, zonas de influencia, corredores, coordenadas de cada corredor, elementos pasivos y robots. Luego se inicia el juego, que da paso al control de turnos e ingresos y salidas de los jugadores.



Figura 20. Ingresar Jugador

Al ingresar un usuario al juego, se verifica si ya se encuentra registrado, si es así se consulta la información del jugador, sus niveles y el historial de movimientos. Si el usuario no se encuentra registrado, se registra como usuario y como jugador, luego se sitúa en el laberinto y se le asigna el turno de jugada.



Figura 21. Controlar Turnos

Cuando el usuario realiza un movimiento, se debe verificar el estado del jugador en la zona de influencia, si existe un jugador en la misma zona que tiene el turno, el usuario debe esperar el turno. Si se le asigna el turno al usuario se consulta la coordenada siguiente para verificar el estado y si se encuentra ocupado por un activo agresivo, si es así se genera una pelea, y finalmente se registra la información del perdedor y el ganador, aumentando al ganador el puntaje y actualizando el estado del perdedor, reasignando los elementos pasivos que poseía.

1. **Diagrama De Concurrencia**

A continuación se presenta el diagrama de concurrencia para el laberinto. Como se presentó en el diagrama de descomposición anterior, todas las llamadas que se realizan desde el cliente se procesan a través del componente **Master**, por lo cual, la primera decisión que se presenta es recibir la petición y que sea procesada asíncronamente para que el proceso sobre el cual se ejecuta el componente, continúe escuchando para recibir nuevas peticiones que vienen de los clientes.

Así mismo, las responsabilidades de componentes como chat, comunicación, seguridad y persistencia se ejecutan sobre un proceso diferente al proceso sobre el cual se ejecuta el componente Master. De esta manera se evita que el componente Master bloquee por mucho tiempo su proceso de ejecución y pueda atender múltiples peticiones de manera concurrente.

Por otra parte, en el cliente es suficiente ejecutar los componentes y sus responsabilidades dentro del mismo proceso.



Figura 22. Diagrama de concurrencia

1. **Lecciones Aprendidas**

* Usar diagramas ad-hoc para representar el contexto del problema funciona de una mejor manera que representarlo por diagramas o notaciones estándares o convencionales.
* Es muy importante definir un método de selección de diseño dentro del grupo, cada uno de los integrantes propuso opciones diferentes las cuales fueron analizadas y después sometidas a votación, con lo cual se seleccionó el diseño que es presentado.

1. **Conclusiones**

* En el desarrollo del análisis de este problema fue necesario crear vistas del juego y apoyarse con diagramas de estado y actividad que representaran las posibles acciones del jugador, para poder mostrar un escenario completo del juego en cada posible jugada, esta aproximación pretende representar los estados del jugador, el sistema y como se modifica tras las posibles jugadas.
* El diseño de un sistema no es un proceso definido estricta y formalmente, sino que depende bastante del contexto en el cual se encuentre el problema. Por ejemplo, en el caso particular del ejercicio del laberinto no existía cierta información que quizás en el diseño de otros sistemas debería ser no solo explicita sino también necesaria, por ejemplo, los actores, los stakeholders, los atributos de calidad o el proceso que se está tratando de modelar. Por este motivo, los entregables de cada análisis no necesariamente son los mismos ni tienen que desarrollarse en el mismo orden, sino que van determinados por el sistema que se quiere analizar.
* Los artefactos generados para el análisis y diseño de un sistema dependen directamente del sistema que se está analizando, no se pueden forzar artefactos que no corresponden y en ocasiones es necesario definir nuevos diagramas y convenciones para poder transmitir la idea.