**Introducción**

***Sobre este documento***

En esta sección se describe la especificación de los requerimientos para el sistema “Walkers of thecity”, juego multijugador en línea desarrollado como trabajo práctico final de las materias Programación Concurrente e Ingeniería del Software. Se provee una descripción de alto nivel del sistema, mostrando los requerimientos de usuario y de sistema y haciendo una distinción entre los requerimientos funcionales y no funcionales del mismo.

***Glosario***

A continuación se listan una serie de términos con sus respectivas definiciones que serán utilizados con frecuencia a lo largo del documento. Se recomienda familiarizarse con los términos para evitar confusiones al leer el presente.

Partida: Una instancia del juego compuesta por una lista de jugadores que participan en ella.

Partida creada: (Createdgame) Es un estado en el que se puede encontrar una partida. Es una partida no comenzada pero lista para iniciar, cuyos parámetros iniciales han sido determinados por el Game Master.

Crear partida: (Creategame) Proceso para instanciar una partida creada

Partida iniciada: (Startedgame) Es un estado en el que se puede encontrar una partida. Partida creada que ha comenzado a desarrollarse.

Iniciar partida: (Startgame) Proceso para instanciar una partida iniciada.

Unirse a una partida: (Joingame) Dada una partida creada un jugador pasa a formar parte de la lista de jugadores que van a participar de dicha partida cuando se inicie.

Jugador: (Player) Un participante que desarrolla el juego dentro de una partida desde un client.

Game Master: Jugador que ha determinado los parámetros iniciales de una partida (creador de una partida). Es el primer jugador que se une a la partida.

Client: Subsistema de software perteneciente al sistema “Walkers of thecity” que hace las veces de cliente en un esquema cliente-servidor proveyendo conectividad con el server.

Server: Subsistema de software perteneciente al sistema “Walkers of thecity” que hace las veces de servidor en un esquema cliente-servidor proveyendo toda la información necesaria, el estado y la lógica del juego.

Graphic Interface: Subsistema de software perteneciente al sistema “Walkers of thecity” cuya función es mostrarle al jugador el estado del juego y proveer un método de acceso para cambiar dicho estado.

Mapa: (Mapa de la ciudad) Mapa virtual que los jugadores deben recorrer. Está compuesto por calles, veredas, autos y repositorios.

Repositorio: Punto específico del mapa a donde el jugador debe llevar la palabra encontrada en la sopa de letras.

***Descripción general del sistema***

El sistema “Walkers of thecity” es un juego multijugador en línea que se desarrolla en partidas creadas por los usuarios.

Cada partida es básicamente una carrera entre los jugadores participantes, que se compone de dos etapas que deben completarse en orden. La primera etapa consiste en buscar una palabra determinada en un sopa de letras. Una vez completada la primera etapa, el jugador accede a la segunda que consiste en llevar la palabra encontrada a través de una ciudad hasta un repositorio. La ciudad consiste de un mapa transitado por autos y por los demás jugadores que hayan completado la primera etapa. Una vez completada esta última etapa el juego termina y el jugador debe esperar a que el resto termine o se cumpla el tiempo límite.

El ganador será el primero en llevar su palabra al repositorio.

**Definición de requerimientos de usuario**

***Requerimientos iniciales***

A continuación se listan una serie de requerimientos de usuario que son tomados como punto de partida para el desarrollo del sistema “Walkers of thecity”.

Dichos requerimientos son el producto de sucesivas entrevistas con el cliente, el Ingeniero Orlando Micolini, quien firmó, junto con nosotros los desarrolladores, un documento que detalla la misma al haber llegado a un acuerdo.

1. Todo usuario del sistema debe ser capaz de conectarse al servidor para crear la partida.
2. Al crearse la partida debe esperarse que se unan la cantidad de jugadores que haya establecido el usuario que creó la partida.
3. La partida iniciará cuando: se hayan unido tantos jugadores como lo indicó el Game Master al crear la partida, o cuando el Game master lo decida.
4. Al comenzar la partida el tiempo general de juego comienza a correr, se le entrega a cada jugador la palabra a buscar, las coordenadas del repositorio y la sopa de letras.
5. El tiempo máximo que puede durar una partida es de 10 minutos.
6. Debe existir un método por el cual el usuario pueda señalar o indicar al sistema la posición de la palabra que le corresponde buscar.
7. Cuando un jugador resuelve correctamente la de la sopa de letras se lo debe ubicar en el mapa y mostrarselo.
8. Los usuarios deben poder moverse a través del mapa y, solo en las esquinas, poder cruzar la calle hacia una de las 2 direcciones posibles.
9. En el mapa los semáforos y vehículos deben ser controlados por el sistema, cumpliendo que el 5% de los autos no respetan a los semáforos, pudiendo chocar a los peatones (usuario-jugador)
10. En el mapa Los cruces de calles se establecen en algunas áreas para dar prioridad a los autos (80%) y en otras para dar prioridad a los peatones (20%)
11. Si un jugador es atropellado debe comenzar en el punto de inicio correspondiente en el mapa.
12. Los autos no colisionan entre sí.
13. Los autos son 50, estos tienen una velocidad al menos 10 veces superior a los peatones, y en las boca calles pueden seguir de largo, doblar derecha e izquierda
14. No puede haber inanición.
15. Al llegar un jugador al repositorio termina su partida, quedándose en una pantalla de espera con la tabla de posiciones hasta el momento, y marcando su tiempo con respecto al tiempo general de juego.
16. El ganador es el que haya llevado la palabra al repositorio en el menor tiempo, y las posiciones se van ocupando a medida que los jugadores entregan las palabra en el repositorio.
17. Una vez llegados a los 10 min de tiempo general de juego, la partida termina instantáneamente para todos los jugadores. Conformándose la tabla general de juego final y mostrándola a todos los jugadores .

**Especificación del sistema**

***Diagrama de casos de uso***



Para completar y extender la información que proporciona el anterior diagrama se han confeccionado las tarjetas de casos de uso para los mostrados.

Las tarjetas de casos de uso se presentan en la siguiente sección.

***Tarjetas de casos de uso***

1) Caso de uso: Unirse a la partida

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Unirse a partida |  |
| **Actor:** | Usuario |  |
| **Descripción:** | el usuario se agrega a una partida creada y queda a la espera de empezar la misma |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  | 1. El usuario peticiona unirse a una partida aleatoria |  |
|  |  | 2. Realiza la conexión necesaria |
|  |  | 3.Le muestra al usuario la partida con la conexión establecida. o el error en la conexión |
| **Precondición:** |  |  |
| **Poscondición:** | Usuario con informe de conectividad y a la espera del inicio de la misma |  |

2) Caso de uso: Crear la partida

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Crear partida |  |
| **Actor:** | Usuario |  |
| **Descripción:** | El usuario crea la partida |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  | 1.Se conecta al servidor |  |
|  | 2.Solicita crear partida determinando en esta solicitud la cantidad de jugadores máxima y otros parametros |  |
|  |  | 3.Crea la partida con sopa de letras, mapa, e iniciando tiempo general de juego en 0 min 0 seg |
|  | 4.Espera por otros jugadores. |  |
| **Flujo alternativo:** | En paso 4, si se completa el numero de usuarios conectados, el sistema inicia la partida,enviando nombre a buscar, mapas y sopas de letras a los clientes. pone a correr el contador |  |
| **Precondición:** | El sistema tiene un servidor habilitado |  |
| **Poscondición:** | Partida creada y espera por otros jugadores, o bien partida iniciada |  |

3) Caso de uso: Iniciar la partida

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Iniciar partida |  |
| **Actor:** | Usuario |  |
| **Descripción:** | El usuario que creo inicia la partida. |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  | 1.Solicita iniciar partida |  |
|  |  | 2.El servidor inicia la partida, enviando nombre a buscar, mapas y sopas de letras a los clientes. pone a correr el contador |
| **Precondición:** | El usuario creo la partida |  |
| **Poscondición:** | Comienzo del juego |  |

4) Caso de uso: Abandonar la partida

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Abandonar partida |  |
| **Actor:** | Usuario |  |
| **Descripción:** | El usuario en medio de una partida, decide abandonarla |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  | 1. El usuario realiza el abandono a una partida |  |
|  |  | 2. Marca registro de abandono del jugador . |
|  |  | 3.Realiza desconexión del servidor |
|  |  | 4. Muestra pantalla de abandono de partida |
| **Precondición:** | Partida creada y en ejecución |  |
| **Poscondición:** | Usuario con informe de abandono. |  |

5) Caso de uso: Jugar con sopa de letras

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Jugar con sopa de letras |  |
| **Actor:** | Usuario |  |
| **Descripción:** | El usuario busca el nombre en la sopa de letras y lo manda al servidor |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  | 1.El usuario busca el nombre en la sopa de letra. |  |
|  | 2.Entrega de las posiciones inicial y final donde se encuentra el nombre encontradas. |  |
|  |  | 3.validez de respuesta, si es incorrecta vuelta al punto 2. , si es correcta , lo lleva a ciudad |
| **Precondición:** | Partida creada e iniciada con sopa de letras y nombre a buscar cargado en el cliente |  |
| **Poscondición:** | El usuario pasa a jugar a la ciudad |  |

6) Caso de uso: Jugar en la ciudad

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Jugar en la ciudad |  |
| **Actor:** | Usuario |  |
| **Descripción:** | El usuario se mueve por el mapa de la ciudad para llevar el nombre al repositorio |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  | 1. El usuario se mueve por el mapa |  |
|  |  | 2. Realiza verificación de movimiento y acciones que desencadena |
|  |  | 3.Informa el movimiento del usuario en el mapa |
|  | 4. Vuelta al punto 1 |  |
| **Precondición:** | El usuario completo el juego de la sopa de letra. |  |
|  | Se le muestra el mapa al usuario con su posicion y posicion de repositorio |  |
|  | El usuario se encuentra en el punto inicial del mapa |  |
| **Poscondición:** | Usuario con el informe de posición actual en el mapa |  |

7) Caso de uso: Crear y configurar semáforos

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Completar partida |  |
| **Actor:** | Usuario |  |
| **Descripción:** | El usuario completa la partida, cumpliendo con los objetivos de ambos juegos antes de que transcurra el tiempo límite. |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  | 1. El usuario llega al repositorio |  |
|  |  | 2. Informa al usuario que ha completado la partida satisfactoriamente. |
|  |  | 3. Muestra tabla de resultados parciales. |
| **Precondición:** | El usuario completo el juego de la sopa de letra. |  |
|  | El usuario está jugando en la ciudad. |  |
| **Poscondición:** | Usuario solo viendo la tabla de resultados parciales. |  |

8) Caso de uso: Crear y configurar semáforos.

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Crear y configurar semáforos |  |
| **Actor:** | Semáforos |  |
| **Descripción:** | Se crean dos semáforos relacionados por cada cruce del mapa y se configuran adecuadamente. |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  |  | 1. Se crea un par de semáforos relacionado para un cruce determinado del mapa. |
|  |  | 2. Se lo configura. |
|  |  | 3. Se lo mapea. |
|  |  | 4. Se vuelve al punto 1 hasta que se cumpla la poscondición. |
| **Precondición:** | No se ha iniciado la partida. |  |
| **Poscondición:** | Hay un par de semáforos por cada cruce del mapa. |  |

9) Caso de uso: Control de semáforos por cruce.

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Control de semáforos por cruce. |  |
| **Actor:** | Semáforos |  |
| **Descripción:** | Se controlan los semáforos de a pares en cada cruce del mapa. |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  |  | 1. En cada cruce del mapa el semáforo que estaba en verde se pone a rojo. |
|  |  | 2. El otro semáforo del mismo cruce del mapa se pone en verde. |
|  |  | 3. Cada cierto tiempo se vuelve al punto 1. |
| **Precondición:** | Hay al menos un Usuario jugando sobre el mapa de la ciudad. |  |
| **Poscondición:** |  |  |

10) Caso de uso: Crear y configurar autos.

|  |  |  |
| --- | --- | --- |
| **Nombre:** | Crear y configurar autos |  |
| **Actor:** | Autos |  |
| **Descripción:** | Se crean y se configuran todos los autos del sistema. |  |
| **Flujo Principal:** | **Eventos ACTOR** | **Eventos SISTEMA** |
|  |  | 1. Se crea un auto. |
|  |  | 2. Se le configura la velocidad, posición inicial y otros elementos de importancia. |
|  |  | 3. Se vuelve al punto 1 hasta que se cumpla la poscondición. |
| **Precondición:** | No se ha iniciado la partida. |  |
| **Poscondición:** | Se han creado todos los autos necesarios del sistema. |  |

***Diagramas de actividad***

Diagrama general del sistema:

Especifica el desarrollo del juego desde el comienzo hasta el final.

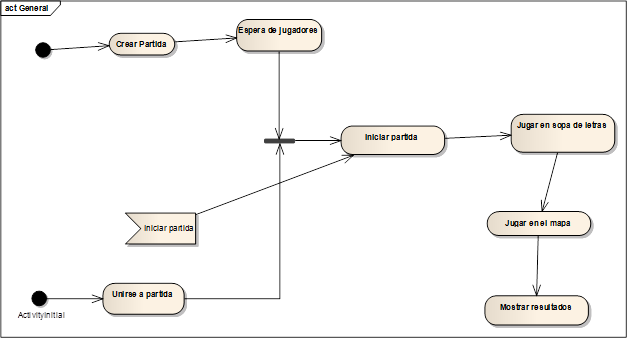


Diagrama de juego sobre el mapa:

Especifica el desarrollo de la segunda etapa del juego que se lleva a cabo sobre el mapa de la ciudad.



***Diagramas de secuencia***

Crear partida / Inicio automático:

Especifica el escenario donde un usuario crea una partida definiendo un número máximo de jugadores. Los jugadores se van uniendo hasta que se alcanza el número máximo permitido (especificado por el Game master), punto en el cual comienzo la partida.



Crear partida / Inicio no automático: Especifica el escenario donde un usuario crea una partida definiendo un número máximo de jugadores. La partida, sin embargo, comienza antes de que se alcance ese número, ya que el Game master dio la orden.



Juego sobre la sopa de letras: Especifica el desarrollo de la primera etapa del juego en donde el jugador debe encontrar una palabra determinada dentro de una sopa de letras.



Movimientos sobre mapa: Especifica el desarrollo de la segunda etapa del juego en donde el jugador se mueve sobre el mapa de la ciudad.



Refresco de interfaz gráfica: Especifica los sucesivos refrescos de la interfaz gráfica que se dan en escenarios donde el jugador está jugando sobre la ciudad, permitiendo que el mismo contenga un estado actualizado del mapa.



Movimiento de un auto: Especifica como el sistema hace que un auto se desplace por la ciudad sin chocar con los demás. Muestra la presencia de monitores que funcionan en base al funcionamiento de una red de Petri.



*Figure:* Dynamic View

**Requerimientos**

***Análisis de requerimientos***

Gracias al análisis de requerimientos que se hizo en las secciones anteriores es posible generar (se descubren nuevos), ordenar y clasificar los mismos, para luego verificar su consistencia (que ningún requerimiento se contradiga con otro) y su validez (que todo el sistema propuesto cumpla con los requerimientos iniciales que impone el cliente) a través de un estudio de los mismos.

A partir de este estudio hemos obtenido los siguientes requerimientos funcionales y no funcionales que se detallan en las siguientes dos secciones.

## Modelo de requerimientos



Requerimientos funcionales



*Figure:* Functional Requirements

Partidas



*Figure:* Partidas

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.1 Crear la partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Todo usuario del sistema debe ser capaz de conectarse al servidor para crear la partida. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.10 Ganador y tabla de posiciones | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| El ganador es el que haya llevado la palabra al repositorio en el menor tiempo, y las posiciones se van ocupan a medida que los jugadores entregan la palabra en el repositorio. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.2 Establecer la capacidad de jugadores que admite la partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Al crearse la partida, el Game Master establece la cantidad maxima de jugadores que admite esa partida. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.3 Cuando se inicia una partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| La partida iniciara cuando el Game Master lo decida o cuando la cantidad de jugadores que se han unido a la partida alcance el limite maximo establecido para la misma. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.4 Que ocurre al iniciar una partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Cuando se inicia una partida el tiempo general de juego comienza a correr, se le entrega a cada jugador la palabra a buscar, la sopa de letras y las corrdenadas del repositorio. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.5 Unirse a una partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Todo usuario del sistema debe ser capaz de conectarse al servidor para unirse a una partida creada. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.6 Abandonar una partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Todo usuario del sistema debe contar con la posibilidad de retirarse de una partida en cualquier momento sin interferir con el juego de los demas jugadores de la misma partida. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.7 Que ocurre cuando un jugador termina una partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Cuando un jugador termina una partida se le muestra al mismo la tabla de resultados parciales hasta el momento y se la mantiene actualizada. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.8 Cuando un jugador termina una partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Un jugador termina una partida cuando completa la segunda etapa del juego, llevando la palabra al repositorio correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.1.9 Cuando finaliza una partida | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Una partida finaliza completamente cuando todos los jugadores han terminado la partida o cuando haya transcurrido el tiempo general de juego. | | |

Ciudad



*Figure:* Ciudad

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.2 Responsabilidades del sistema | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*High | *Difficulty:*High |
| *Phase: 1.0* | *Version: 1.0* |  |
| En el mapa los semaforos y veiculos deben ser controlados por el sistema. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.1 Interfaz | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Los usuarios deben poder moverse a traves del mapa y, solo en las esquinas, poder cruzar la calle hacia una de las 2 direcciones posibles. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.10 Cuando se gana la etapa | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Al llegar un jugador al repositorio se completa la segunda etapa satisfactoriamente, y por el REQ 1.1.8 el jugador termina la partida. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.11 Cuando un jugador es atropellado | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*High | *Difficulty:*High |
| *Phase: 1.0* | *Version: 1.0* |  |
| La situacion de choque entre un jugador y un auto se da cuando, en un instante dado, un auto pasa por una posicion que estaba siendo ocupada por un jugador (roba el recurso) o un jugador intenta pasar por una posicion que esta siendo ocupada por un auto (intento de robo de recurso). | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.3 Prioridad en los cruces | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| En un 80% de las esquinas existentes, cuando un semaforo esta en verde en un instante dado, los autos pueden pasar por una posicion que esta siendo ocupada por un jugador (puede robarle el recurso) y en el 20% restante esto no se permite. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.4 Autos / Autos | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*High |
| *Phase: 1.0* | *Version: 1.0* |  |
| El sistema debera controlar que los autos no colisionen entre si. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.5 Autos / Semaforos | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| El 5% de los autos no respetan a los semaforos, pudiendo chocar a los peatones. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.6 Que ocurre cuando un jugador es atropellado | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*High | *Difficulty:*Low |
| *Phase: 1.0* | *Version: 1.0* |  |
| Si un jugador es atropellado debe comenzar en el punto de inicio correspondiente en el mapa. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.7 Cantidad de autos | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Low |
| *Phase: 1.0* | *Version: 1.0* |  |
| El sistema debera generar exactamente 50 autos. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.8 Velocidad de autos | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Low | *Difficulty:*Low |
| *Phase: 1.0* | *Version: 1.0* |  |
| El sistema debera controlar que los autos tengan una velicodad al menos 10 veces superior a los peatones. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.3.9 Movimientos de los autos | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*High | *Difficulty:*High |
| *Phase: 1.0* | *Version: 1.0* |  |
| En cada curce de calle del mapa los autos pueden doblar a la izquierda o a la derecha o seguir de largo. El sistema sera quien decida el camino que siguen. | | |

Sopa de letras



*Figure:* Sopa de letras

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.2.1 Interfaz | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*High | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Debe existir un metodo por el cual el usuario pueda señalar o indicar al sistema la posicion de la palabra que le corresponde buscar. Ademas de un sistema a traves del cual el jugador pueda visualizar la sopa de letras. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.2.2 Que ocurre al resolver la sopa de letras | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Cuando un jugador resuelve correctamente la sopa de letras pasa a la siguiente estapa del juego. Se lo debe ubicar en el mapa y mostrarselo. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.2.3 Contenido de la sopa de letras | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*High | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| La sopa de letras que se le proporciona al jugador al iniciar la partida debe contener dentro de ella el palabra que el jugador tiene que encontrar. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 1.2.4 Cuando se resuleve la sopa de letras | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| La sopa de letras se resuelve cuando el jugador indica donde se encuentra la palabra que está buscando. | | |

#### Cliente



*Figure:* Cliente

|  |  |  |  |
| --- | --- | --- | --- |
| Interfaz grafica | | | |
| *«Functional»* | *Status:* Proposed | *Priority:* Medium | *Difficulty:* Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| El cliente SOLO muestra la interfaz grafica y no tiene nada de la logica del juego ni los datos del mismo. | | |

#### Servidor



*Figure:* Servidor

|  |  |  |  |
| --- | --- | --- | --- |
| Control de logica de autos y semaforos | | | |
| *«Functional»* | *Status:* Proposed | *Priority:* Medium | *Difficulty:* Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| Todo el control de la la logica de los autos y los semaforos se ejecutara en el servidor. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Resguardo de datos de todo el juego en general | | | |
| *«Functional»* | *Status:* Proposed | *Priority:* Medium | *Difficulty:* Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| El servidor sera el responsable de contener y mantener actualizados todos los datos del juego, ya sea los datos de partidas, de la ciudad y de la sopa de letras. | | |

Requerimientos no funcionales



*Figure:* Non-Functional Requirements

Concurrencia



*Figure:* Concurrencia

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 2.1.2 Inanicion | | | |
| *«Performance»* | *Status:*Proposed | *Priority:*High | *Difficulty:*High |
| *Phase: 1.0* | *Version: 1.0* |  |
| No puede haber inanicion entre hilos. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 2.1.3 Interbloqueos | | | |
| *«Functional»* | *Status:*Proposed | *Priority:*High | *Difficulty:*High |
| *Phase: 1.0* | *Version: 1.0* |  |
| Para que el sistema funcione correctamente no debe haber interbloqueos entre hilos. | | |

General



*Figure:* General

|  |  |  |  |
| --- | --- | --- | --- |
| REQ 2.1.1 Duracion maxima de una partida | | | |
| *«Performance»* | *Status:*Proposed | *Priority:*Medium | *Difficulty:*Medium |
| *Phase: 1.0* | *Version: 1.0* |  |
| El tiempo general de juego nunca puede ser mayor a 10 minutos. | | |

***Matriz de trazabilidad entre casos de uso y requerimientos***

Para simplificar la matriz se asignan los siguientes acrónimos a los casos de uso:

* Crear partida: UC1
* Unirse a partida: UC2
* Iniciar partida: UC3
* Abandonar partida: UC4
* Jugar con sopa de letras: UC5
* Jugar en la ciudad: UC6
* Completar partida: UC7
* Definir parámetros de partida: UC8
* Indicar dónde se encuentra la palabra: UC9
* Moverse por la ciudad: UC10
* Ver tabla de resultados: UC11

Partidas:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **REQ** | **UC1** | **UC2** | **UC3** | **UC4** | **UC5** | **UC6** | **UC7** | **UC8** | **UC9** | **UC10** | **UC11** |
| **1.1.1** | **x** |  |  |  |  |  |  |  |  |  |  |
| **1.1.2** | **x** |  |  |  |  |  |  | **x** |  |  |  |
| **1.1.3** |  |  | **x** |  |  |  |  | **x** |  |  |  |
| **1.1.4** |  |  | **x** |  |  |  |  |  |  |  |  |
| **1.1.5** |  | **x** |  |  |  |  |  |  |  |  |  |
| **1.1.6** |  |  |  | **x** |  |  |  |  |  |  |  |
| **1.1.7** |  |  |  |  |  |  | **x** |  |  |  | **x** |
| **1.1.8** |  |  |  |  |  |  | **x** |  |  |  |  |
| **1.1.9** |  |  |  |  |  |  | **x** |  |  |  |  |
| **1.1.10** |  |  |  |  |  |  | **x** |  |  |  | **x** |

Sopa de letras:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **REQ** | **UC1** | **UC2** | **UC3** | **UC4** | **UC5** | **UC6** | **UC7** | **UC8** | **UC9** | **UC10** | **UC11** |
| **1.2.1** |  |  |  |  | **x** |  |  |  | **x** |  |  |
| **1.2.2** |  |  |  |  | **x** |  |  |  |  |  |  |
| **1.2.3** |  |  |  |  | **x** |  |  |  |  |  |  |
| **1.2.4** |  |  |  |  | **x** |  |  |  | **x** |  |  |

Ciudad:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **REQ** | **UC1** | **UC2** | **UC3** | **UC4** | **UC5** | **UC6** | **UC7** | **UC8** | **UC9** | **UC10** | **UC11** |
| **1.3.1** |  |  |  |  |  | **x** |  |  |  | **x** |  |
| **1.3.2** |  |  |  |  |  | **x** |  |  |  |  |  |
| **1.3.3** |  |  |  |  |  | **x** |  |  |  |  |  |
| **1.3.4** |  |  |  |  |  | **x** |  |  |  |  |  |
| **1.3.5** |  |  |  |  |  | **x** |  |  |  |  |  |
| **1.3.6** |  |  |  |  |  | **x** |  |  |  |  |  |
| **1.3.7** |  |  |  |  |  | **x** |  |  |  |  |  |
| **1.3.8** |  |  |  |  |  | **x** |  |  |  |  |  |
| **1.3.9** |  |  |  |  |  | **x** |  |  |  |  |  |
| **1.3.10** |  |  |  |  |  | **x** | **x** |  |  |  |  |
| **1.3.11** |  |  |  |  |  | **x** |  |  |  |  |  |

***Matriz de trazabilidad entre requerimientos***

(ver aparte)

**Arquitectura preliminar del sistema**

