Capítulo 3

En este capítulo se describe el funcionamiento de los componentes de la librería, además de la arquitectura de la aplicación propuesta para demostrar la librería creada. A la vez se analiza a profundidad el funcionamiento de algoritmos novedosos que conforman la estructura del programa. Los algoritmos mencionados son explicados en el orden de aparición en la aplicación.

3.1 Algoritmos propuestos

Se describen a continuación las distintas operaciones que puede realizar el usuario, para lo cual se necesita implementar algoritmos, algunos de ellos se encuentran basados en teorías matemáticas bien fundamentadas.

La herramienta debe brindar al usuario una serie de opciones sobre dónde desea que se realice la partida. Con este fin se le provee una pantalla inicial con la lista de las ciudades importantes que cuentan con este servicio. Una vez que el usuario haya realizado su selección, la herramienta puede seleccionar aleatoriamente el punto de inicio, en caso que sea una partida de un solo jugador o la ubicación del objetivo en caso contrario. Después se suma o resta una cantidad predefinida por el sistema o definida por el usuario, la cual es la distancia de la meta a la cual se encontrará cada jugador. Se puede determinar el número de jugadores que aparecerán, para lo que el sistema determina las nuevas posiciones de los jugadores de forma circular, garantizando que se encuentren a la misma distancia pero en distinta posición.

La posibilidad de bloquear rutas a los jugadores para poner mayores dificultades es posible y fue considerada, sin embargo no hay forma de saber si una ruta es la única forma de llegar a la meta. Es decir si se bloquea una calle no hay forma de saber que esa calle es la única vía que permite acceso a una cierta posición donde se encuentra la meta. En su

lugar se recurre a simulaciones de tráfico en las calles. El tráfico corresponde a un punto en el mapa con la característica que en él y sus alrededores un usuario avanzará una distancia menor a la normal.

En un principio se tenía contemplado agregar disparos a la aplicación con el fin de detener al rival; sin embargo, por estar en un contexto de aplicación formal y en un momento en el tiempo en el que se busca el aprendizaje colaborativo no fue implementado.

3.1.1 Movimientos

Una vez que se conocen los elementos que conforman el libre movimiento por las calles, se infiere que se necesitan dos métodos básicos necesarios para que el usuario pueda tener el control. El primer método es avanzar y el segundo es girar.

Avanzar consiste en recorrer una distancia previamente establecida hacia el lugar donde el usuario está volteando. Este desplazamiento requiere el valor actual de la orientación para moverse hacia esa dirección en particular, para ello es importante tener almacenadas las variables actuales, especialmente las que se refieren a la orientación. En caso que la dirección hacia la cual el usuario quiera ir no sea válida, el sistema no se actualiza sino que se queda en la última posición válida. La figura 3.1 demuestra esta habilidad mostrando en la parte superior la posición del usuario casi a mitad de una calle e inmediatamente después se encuentra a escasos pasos de la esquina representando un movimiento.

Figura 3.1 Representación gráfica de movimiento.

El segundo método es llamado girar, el cual cambia el punto de vista del usuario hacia la orientación que él desee, dándole 360 grados de libertad. Girar es análogo a una persona que se encuentra en medio de un cruce de calles y desea colocarse de pie mirando hacia la calle que se encuentra a su costado sin moverse del lugar donde se encuentra. En la figura 3.2 se presenta un giro a la derecha, donde la imagen de la izquierda representa un panorama viendo exactamente hacia el norte, en la imagen de la derecha se presenta la misma imagen pero vista desde un ángulo distinto, que demuestra el giro.

Figura 3.2 Representación gráfica de giro a la derecha.

Adicionalmente a estos métodos de movimiento básicos la librería es capaz de comparar ubicaciones para determinar si uno o más usuarios se encuentran en la misma posición, significando que hubo un choque entre ellos, dando opción de inmovilizar a los usuarios. Es necesario también un método de comparación que establezca si uno o más jugadores se encuentren cercanos unos a otros, pudiendo manipular la distancia del radio que establece esta cercanía.

Otra capacidad es dar pasos laterales, esto significa moverse hacia la derecha o hacia la izquierda sin necesidad de cambiar la orientación hacia la cual el jugador está mirando. El número de pasos que el usuario recorre por cada movimiento es también controlado por la librería, al igual que la distancia que recorre; por ejemplo un usuario puede avanzar 7 pasos hacia adelante, avanzando 5 metros por cada paso para recorrer una distancia total de 35 metros.

3.2 Interfaz

Con el objeto de demostrar las capacidades de WITS se crea una aplicación, su interfaz consiste en todo lo que el usuario ve y a través de lo cual se comunica y esto comienza con el menú principal a través del cual se puede elegir el tipo de partida. Independientemente de la opción seleccionada este envía al menú de selección de ciudad, mediante el cual se puede seleccionar la ciudad deseada sobre la cual se realizará la partida. Debe de haber para cada

jugador un “radar” en la esquina superior derecha para poder ubicarse con mayor facilidad y ver la posición de los demás jugadores, de esta forma los usuarios pueden planear nuevas rutas en base a la posición de los demás. En la figura 3.3 se muestra un ejemplo de esto en donde el jugador siempre será mostrado al centro mientras que la posición de los contrincantes se encuentra denotada de otro color alrededor del jugador.

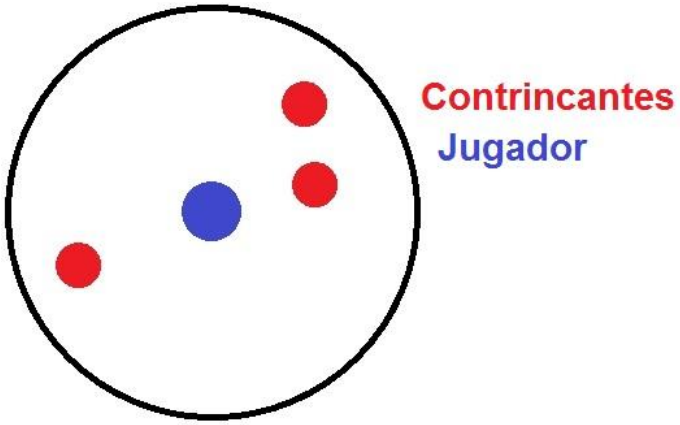

Figura 3.3 Ejemplo de radar.

En el centro de la pantalla debe haber una imagen que muestre los alrededores de la posición donde el usuario se encuentre, denotando gráficamente su posición geográfica específicamente en las calles de la ciudad seleccionada, este componente se llama mapa.

Deben existir lugares específicamente delimitados e inamovibles con los cuales se pueda controlar la posición geográfica, estos lugares son llamados zonas de control. Con el fin de evitar que la zona de control sea igual al mapa, lo que causaría que el usuario retire la mano de la pantalla después de cada movimiento para poder observar su nueva ubicación, se colocan en lugares distintos de la pantalla.

3.3 Arquitectura de aplicación.

En esta sección se describe detalladamente la forma en que los componentes de la aplicación están ligados entre sí. Este es sólo un tipo de arquitectura usado específicamente para esta aplicación, sin embargo la librería permite la creación de distintas composiciones

dependiendo del gusto y necesidades del programador.

3.3.1 Arquitectura multicontacto

En el capítulo dos se presentó una manera en que el sistema podría basarse en cuanto al multicontacto se refiere. Durante el proceso de construcción fue evidente que dicha arquitectura es correcta ya que separa de manera adecuada el control de cada uno de los elementos y además se ha observado que la mayoría de las aplicaciones siguen el mismo patrón de diseño en este aspecto. Sin embargo para la versión multijugador esta arquitectura no cuenta con los elementos suficientes, por lo tanto la arquitectura de la aplicación, aunque es muy parecida cuenta con otros elementos (Figura 3.4).

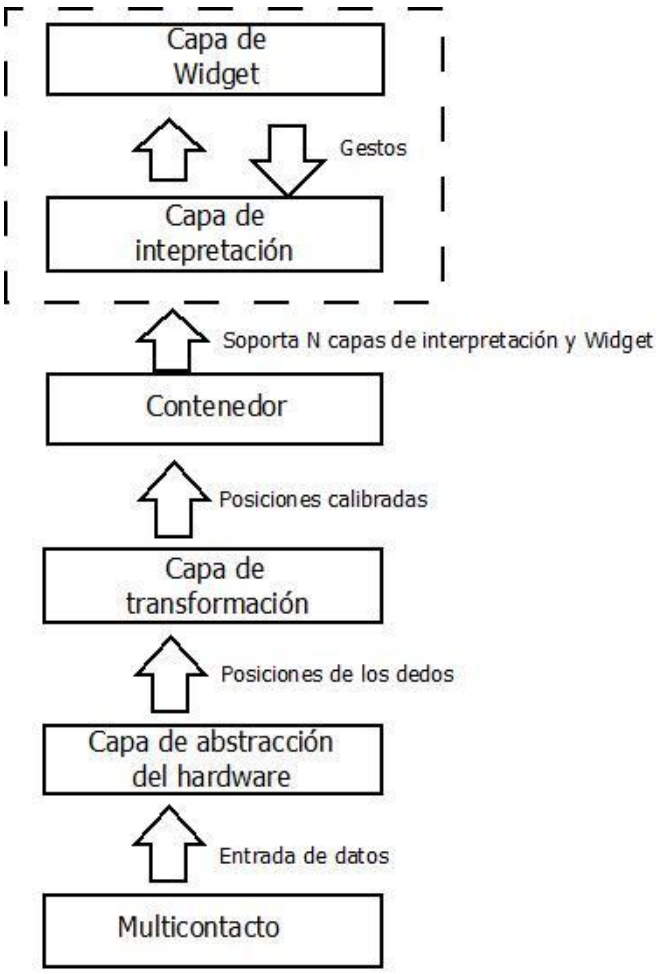

Figura 3.4-Diagrama de la arquitectura multicontacto.

De las capas existentes en la arquitectura, la capa de abstracción del hardware es la que

menos preocupación da al programador porque se encuentra previamente instalado en el dispositivo multicontacto. Su trabajo es detectar cualquier contacto o movimiento que suceda dentro de los límites de la superficie y reportarlos a la siguiente capa, es decir es el rastreador. Aunque es capaz de detectar movimiento no lo hace de manera minuciosa, sino que pasa los valores a la siguiente capa para que se encargue de refinar las coordenadas.

La capa de abstracción del hardware pasa a la capa de transformación los datos registrados. En esta capa no se está hablando de hardware sino de software. Hay varios componentes capaces de manejar la información obtenida por la capa anterior, los que son conocidos como librerías dinámicas, las cuales es necesario instalar en el dispositivo multicontacto. Esta capa recibe los datos de la capa anterior, es decir está consciente que ocurrió un evento en la pantalla y le corresponde a esta capa ubicar el punto exacto en el que sucedió, si sufrió algún cambio y en qué momento terminó, guardando además los datos del primero y último punto de contacto registrado.

El contenedor es un elemento agregado sobre el cual se colocan cualquier cantidad de capas de interpretación y de *widget* y es la primera de las capas que se encuentra a cargo del programador. No tiene restricción en cuanto al número de capas que se agreguen, sin embargo evidentemente entre más se agreguen cada una ocupa un menor espacio en la superficie hasta llegar a ser imperceptible para el ojo humano. Al ser una misma capa sobre la que se encuentran los componentes permite que todos ellos puedan ser tocados. Cuando sea una partida entre varios jugadores esta capa es la encargada de distribuir a los jugadores en la ciudad escogida mediante el uso de la librería.

La capa de interpretación consiste en indicar a la aplicación lo que debe hacer una vez que tenga las coordenadas específicas y los movimientos que se realizaron sobre ellas. El funcionamiento de esta capa es mediante eventos que deben ser filtrados para poder asignar tareas al sistema con las coordenadas especificadas en el evento. Es decir, esta capa se

convierte en interpretadora de los gestos hechos en la pantalla.

La siguiente capa es la de *widget*, esta es la capa de mayor importancia desde el punto de vista del usuario, es donde va a ver reflejados los cambios de los gestos que realizó sobre la superficie, por tanto debe ser agradable a la vista y procurando que los elementos que la conforman sean totalmente intuitivos. De la misma forma los resultados de las acciones deben desplegarse al usuario en esta capa de forma clara e inmediata para una experiencia fluida en la aplicación sin que esto signifique descuidar las capas anteriores. Es de suma importancia mantener transparencia para el usuario, es decir el usuario no requiere saber los procesos que se iniciaron para poder realizar las acciones deseadas o de dónde se obtuvo la información. Por ello la capa de *Widget* únicamente despliega la información, sin mostrar información de las capas anteriores. Internamente la capa de widget recibe órdenes de la capa de interpretación sobre qué es lo que tiene que desplegar en un momento dado sin resolver cosas por sí misma sino que depende de la capa anterior. Hecho el cambio, nuevamente espera órdenes de la capa de interpretación para poder empezar de nuevo según el tiempo requerido del usuario.

3.4 Arquitectura de componentes

La librería consiste de dos clases, una que modela la ubicación y otra que controla las distintas ubicaciones simultáneamente. Es sabido que a cada usuario le corresponde una zona de control, un mapa y un radar. Los gestos que realice el usuario sobre su zona de control hacen que la ubicación actual en el mapa cambie, es decir afecta el estado de su ubicación lo cual se muestra en el mapa; además, la ubicación actual del usuario se representa en el radar. En caso de ser multijugador, la posición de los demás jugadores también se muestra en el radar.

Si cada jugador se representa por una posición en el mapa, de igual forma que la

coordenada del punto objetivo, cada uno de estos es controlado por un objeto de tipo "controlador" (Figura 3.5). El controlador posee además una lista con los distintos puntos donde hay tráfico, los cuales también son una posición en el mapa y pueden ser manejados como tal.

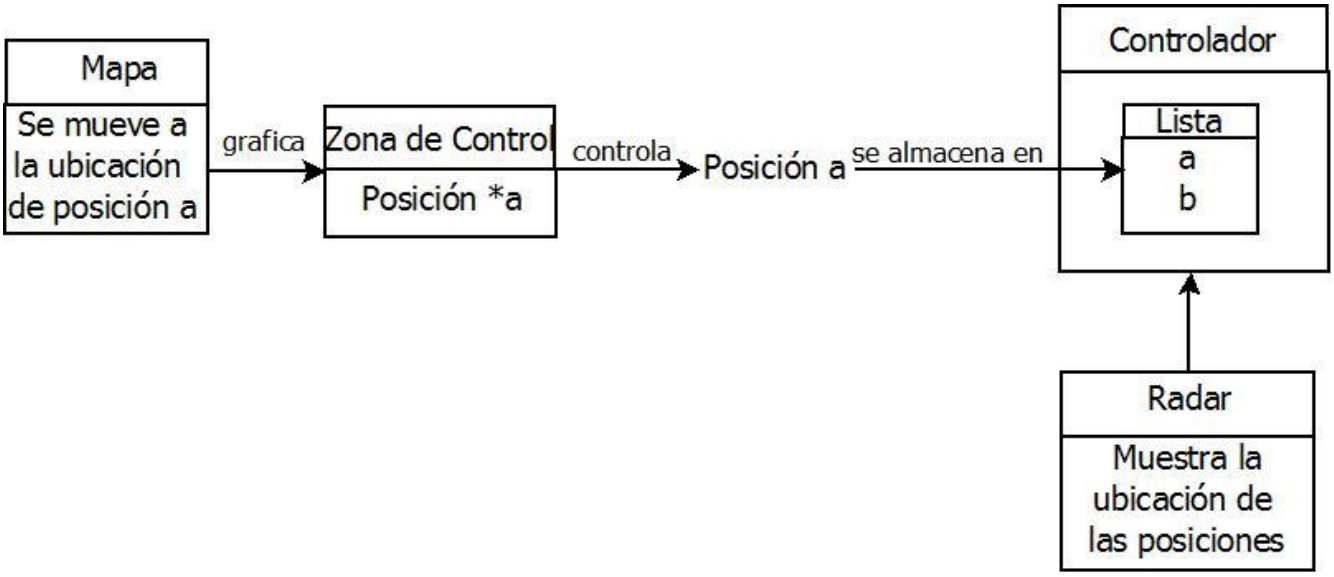

Figura 3.5-Interacción de componentes

La librería tiene una facilidad de convertirse fácilmente a cualquier tipo de arquitectura, basta con agregarla a la aplicación que se desee programar dando la opción de ser usada en aplicaciones que no requieran de multicontacto.

3.5 Prototipo de baja fidelidad

Se realizó un prototipo de baja fidelidad simulando el funcionamiento y la interfaz gráfica del sistema utilizando hojas de papel y dibujos sin detallar mucho en la apariencia de la misma. Se presentó este prototipo a individuos de distintas disciplinas que se encuentran en el grupo hacia el que está orientada la aplicación.

A grandes rasgos la interfaz gráfica les es entendible a los usuarios. Las zonas de control al tener la mano dibujada sí cumple el objetivo de brindar la idea al usuario que el control debe ser con la palma de la mano, sin embargo no resulta tan obvio el que la mano

se tiene que colocar exclusivamente en esa área, sino que colocaban la mano en el centro de la pantalla.

Aquellos que no suelen frecuentar los juegos de video, tuvieron cierta confusión al momento de entrar a una partida multijugador y ver que la pantalla se divide. Este hecho causa que estos usuarios pierdan unos segundos en ubicar su pantalla y razonar lo que está pasando, pero después de este tiempo les parece natural. Por el contrario a los usuarios que tienen experiencia en juegos de video esta división les parece inmediatamente familiar y no tienen problema para usarlo.

El sistema de navegación les parece correcto a los usuarios, acorde al contexto y fácil de utilizar. El radar en la esquina les parece bastante útil y colocado en una buena posición que no afecta la vista del lugar donde se encuentran.

En cuanto a la navegación entre ventanas les parece bastante intuitivo y no tienen problema alguno para navegar y seleccionar las opciones deseadas o incluso regresar a pantallas anteriores para poder configurar el juego.

Los sujetos de estudio dieron algunas recomendaciones, las cuales fueron tomadas en cuenta, entre ellas se encuentran:

- Permitir al usuario seleccionar el lado en el que desee que la zona de control aparezca, dando accesibilidad a los zurdos.
- Mostrar una flecha en el mapa que indique la dirección hacia donde se dirige la nave en el mapa.
- Ofrecer beneficios extras en el transcurso del juego colocados en lugares específicos del mapa por ejemplo un escudo que impida que el contrincante dañe a la nave del jugador que lo tomó.
- Colocación de algún texto que sugiera al usuario que su mano debe ir colocada

en ese espacio para controlar la nave.

- El método que active los disparos sea directamente señalar al contrario, en lugar de que se encuentre también en la zona de control.

3.6 Resumen del capítulo

En este capítulo se presenta la arquitectura a la que la aplicación se atiene, además de la versatilidad con la cual se puede cambiar a otro tipo de arquitectura que incluya multicontacto o no. Se describe también a detalle el funcionamiento de algoritmos nuevos que la librería utiliza y el cómo algunas no pudieron llevarse a cabo de la manera que fueron planeadas originalmente sin embargo esto no afecta el desarrollo de la aplicación.

En los siguientes capítulos se verá a detalle cómo se llevó a la práctica este diseño y algoritmos, para analizar después qué tan efectivos resultaron en base a una evaluación. En el último capítulo se verá las conclusiones sobre este diseño.