Juego de estrategia en tiempo real: Z

Ejercicio Final

Objetivos	● Afianzar los conocimientos adquiridos durante la cursada. ● Poner en práctica la coordinación de tareas dentro de un grupo de trabajo. ● Realizar un aplicativo de complejidad media con niveles aceptables de calidad y usabilidad.
Instancias de Entrega	**Pre-Entrega:** clase 14 (13/06/2017). **Entrega:** clase 16 (27/06/2017).
Temas de Repaso	● Aplicaciones Cliente-Servidor multi-threading. ● Interfaces gráficas con *gtkmm* ● Manejo de errores en C++
Criterios de Evaluación	● Criterios de ejercicios anteriores. ● Construcción de un sistema Cliente-Servidor de complejidad media. ● Empleo de buenas prácticas de programación en C++. ● Coordinación de trabajo grupal. ● Planificación y distribución de tareas para cumplir con los plazos de entrega pautados. ● Cumplimiento de todos los requerimientos técnicos y funcionales. ● Facilidad de instalación y ejecución del sistema final. ● Calidad de la documentación técnica y manuales entregados. ● Buena presentación del trabajo práctico y cumplimiento de las normas de entrega establecidas por la cátedra (revisar criterios en sitio de la materia).

Índice

Introducción

Z es un juego en tiempo real en donde cada jugador comienza con un Fuerte y un grupo de robots y vehículos con el objetivo de destruir los Fuertes enemigos y proteger el propio.
El presente trabajo es una *remake* del juego original de Bitmap Brothers [1] con soporte para multijugador y armado de equipos.

Descripción

Dinámica del juego

Cada jugador comienza con un Fuerte y un grupo de robots y vehículos. El mapa completo está dividido en territorios donde cada jugador comienza como dueño de uno de ellos en donde está su Fuerte, por su puesto.

El objetivo del juego es destruir los Fuertes enemigos o bien destruir todas las unidades enemigas. En dicho caso, el Fuerte del enemigo es automáticamente destruido.

Para alcanzar el objetivo es necesario fabricar robots y vehículos con los cuales combatir al enemigo e capturar nuevos territorios.

Cada territorio tiene una bandera que al ser capturada por un robot o vehículo provoca automáticamente la captura del territorio, todas sus fábricas y estructuras para dicho jugador.
Como caso especial está el territorio donde se encuentra el Fuerte. Su bandera no es accesible ni puede ser tomada por otro jugador a menos que el Fuerte sea destruido primero.

Capturar un territorio no solo habilita al jugador a usar las estructuras del territorio sino que además mejora los tiempos de fabricación de todas las fábricas controladas por el jugador.

Distribuidos por el mapa hay vehículos abandonados que pueden ser capturados por los robots de un jugador.

Como el juego es multijugador es posible armar equipos: la victoria de un equipo se da cuando los Fuertes de los equipos enemigos son destruidos (o todas las unidades enemigas son destruidas).
Las unidades de un jugador no podrán atacar a las unidades ni edificios de otro jugador de su mismo equipo pero en lo que respecta a la captura de un territorio no hay ninguna limitación: un jugador le puede robar el territorio a un jugador amigo.

Tipos de terreno

Tierra / Pradera / Nieve: Tanto los robots como los vehículos pueden desplazarse sobre este tipo de terreno. No hay diferencia entre la Tierra, la Pradera o la Nieve salvo que se muestran distintas imágenes en pantalla.

Agua / Pantano: Los robots pueden desplazarse sobre este tipo de terreno a una velocidad 0.7 veces la

velocidad que tendrían por Tierra. Los vehículos no pueden desplazarse sobre este tipo de terreno. No hay diferencia entre el Agua y el Pantano salvo que se muestran distintas imágenes en pantalla.

Lava: Ninguna unidad puede desplazarse sobre este tipo de terreno.

Carretera / Camino Asfaltado: Tanto los robots como los vehículos pueden desplazarse sobre este tipo de terreno con una velocidad mayor a la que si lo hicieran por Tierra. La velocidad es de 1.5 veces la velocidad que tendrían por Tierra. No hay diferencia entre la Carretera y el Camino Asfaltado salvo que se muestran distintas imágenes en pantalla.

Objetos sobre el terreno

Rocas / Bloques de Hielo: Ninguna unidad puede desplazarse sobre este tipo de terreno. Sin embargo estas obstrucciones pueden ser destruidas por cualquier unidad o vehículo que disponga de un ataque explosivo, liberando el terreno y abriendo paso. No hay diferencia entre las Rocas y los Bloques de Hielo salvo que se muestran distintas imágenes en pantalla.
Los puntos de estructura (o resistencia a los daños) de las Rocas/Bloques de hielo es de 10.

Edificios (Fábricas, Fuertes): Ninguna unidad puede desplazarse sobre este tipo de terreno. Los edificios pueden ser dañados y hasta destruidos por cualquier unidad o vehículo que disponga de un ataque explosivo sin embargo las ruinas del edificio permanecerán ahí y continuarán bloqueando el paso.
Los puntos de estructura (o resistencia a los daños) de los Edificios es de 1000.

Puente de Concreto / Puente de Madera: Es un objeto que está por sobre el Agua/Pantano o la Lava. Tanto los robots como los vehículos pueden desplazarse sobre este tipo de terreno. Como otros objetos los puentes pueden ser dañados y destruidos por cualquier unidad que disponga de un ataque explosivo. Un puente dañado no tiene ningún efecto adicional sobre el juego pero un puente destruido hace que ninguna unidad pueda desplazarse sobre él. No hay diferencia entre el Puente de Concreto y el Puente de Madera salvo que se muestran distintas imágenes en pantalla.
Los puntos de estructura (o resistencia a los daños) de los Puentes es de 1000.

Bandera: Tanto los robots como los vehículos pueden capturar una bandera y desplazarse con ella. Las banderas no pueden ser destruidas.

Edificios

Fuerte: Es el edificio principal que cada jugador tiene. Puede fabricar robots y vehículos. Qué tipos de unidades estará determinado por el nivel de tecnología del fuerte.

Fabrica de Robots: Como lo indica su nombre, este tipo de fábricas permite la construcción de nuevas unidades de a pie o robots. Los tipos de unidades son determinados por el nivel de tecnología de la fábrica.

Fábrica de vehículos: Como lo indica su nombre, este tipo de fábricas permite la construcción de nuevos vehículos. Los tipos de vehículos/unidades son determinado por el nivel de tecnología de la fábrica.

Armamento

Los robots y los vehículos cuentan cada uno de ellos con un armamento propio que determina el tipo de ataque, el daño que producen y la animación de los proyectiles que son lanzados.

Balas:
Daño: 2
Animacion: Ninguna (hacen impacto inmediatamente)
Ataque explosivo: No.

Proyectiles de alto calibre:
Daño: 20
Animación: Los proyectiles se desplazan por el campo desde quien los lanza hasta el lugar del impacto. Al impactar debe mostrarse la animación de una explosión pequeña.
Ataque explosivo: Si.

Misiles:
Daño: 25
Animación: Los misiles se desplazan por el campo desde quien los lanza hasta el lugar del impacto. Al impactar debe mostrarse la animación de una explosión grande.
Ataque explosivo: Si.

Lanzallamas:
Daño: 10
Animación: Se muestra cómo las llamas se desplazan por el campo desde quien las lanza hasta el lugar del impacto.
Ataque explosivo: Si.

Laser:
Daño: 10
Animación: Se muestra como los rayos láser se desplazan por el campo desde quien los lanza hasta el lugar del impacto.
Ataque explosivo: No.

Robots

Los robots son unidades de a pie que se fabrican de a grupos. Cada tipo de unidad cuenta con su propio tipo de armamento, frecuencia de disparo, alcance y puntos de estructura. Todos los robots tienen la misma velocidad de desplazamiento: 4.
Así también cada tipo de unidad requiere de cierto nivel de tecnología mínimo en la Fábrica o Fuerte para ser construido y tiene un tiempo de fabricación base propio.

Gruñido (Grunt):
Armamento: Balas
Frecuencia de disparo: 2 por segundo.
Alcance: 7
Puntos de estructura: 60
Fabricación: de a 3 robots.
Tiempo de fabricación base: 9:35 minutos.
Nivel de tecnología mínima: 1

Psico (Psycho):
Armamento: Balas
Frecuencia de disparo: 10 por segundo.
Alcance: 7
Puntos de estructura: 80
Fabricación: de a 3 robots.
Tiempo de fabricación base: 10 minutos.
Nivel de tecnología mínima: 2

Duro (Tough):
Armamento: Misiles
Frecuencia de disparo: 2 por segundo.
Alcance: 5
Puntos de estructura: 300
Fabricación: de a 2 robots.
Tiempo de fabricación base: 12:22 minutos.
Nivel de tecnología mínima: 2

Piro (Pyro):
Armamento: Lanzallamas
Frecuencia de disparo: 4 por segundo.
Alcance: 6
Puntos de estructura: 100
Fabricación: de a 4 robots.
Tiempo de fabricación base: 14 minutos.
Nivel de tecnología mínima: 4

Francotirador (Sniper):
Armamento: Balas
Frecuencia de disparo: 4 por segundo.
Alcance: 10
Puntos de estructura: 80
Fabricación: de a 3 robots.
Tiempo de fabricación base: 9:35 minutos.
Nivel de tecnología mínima: 3

Láser (Laser):
Armamento: Laser
Frecuencia de disparo: 4 por segundo.
Alcance: 7
Puntos de estructura: 100
Fabricación: de a 5 robots.
Tiempo de fabricación base: 15 minutos.
Nivel de tecnología mínima: 5

Vehículos

Los vehículos son creados de a uno y son conducidos por un robot de tipo Gruñido (Grunt). Pueden existir vehículos sin conductor en el campo que al ser capturados por un robot, este se convierte en su conductor. Cada vehículo cuenta con su propio tipo de armamento, frecuencia de disparo, alcance, puntos de estructura y velocidad base.

Cada tipo de unidad requiere de cierto nivel de tecnología mínimo en la Fábrica o Fuerte para ser construido y tiene un tiempo de fabricación base propio.

Jeep (Jeep):
Armamento: Balas
Frecuencia de disparo: 6 por segundo.
Alcance: 6
Puntos de estructura: 60
Velocidad base: 8
Tiempo de fabricación base: 11 minutos.
Nivel de tecnología mínima: 1

Tanque medio (Medium Tank):
Armamento: Proyectiles de alto calibre
Frecuencia de disparo: 0.5 por segundo.
Alcance: 7
Puntos de estructura: 120
Velocidad base: 5
Tiempo de fabricación base: 18 minutos.
Nivel de tecnología mínima: 3

Tanque ligero (Light Tank):
Armamento: Proyectiles de alto calibre
Frecuencia de disparo: 0.5 por segundo.
Alcance: 6
Puntos de estructura: 80
Velocidad base: 6
Tiempo de fabricación base: 14 minutos.
Nivel de tecnología mínima: 2

Tanque pesado (Heavy Tank):
Armamento: Proyectiles de alto calibre
Frecuencia de disparo: 0.5 por segundo.
Alcance: 8
Puntos de estructura: 180
Velocidad base: 5
Tiempo de fabricación base: 20 minutos.
Nivel de tecnología mínima: 4

Lanzador de misiles móvil (Movile Missile Launcher o MML):
Armamento: Misiles
Frecuencia de disparo: 2 por segundo.
Alcance: 8
Puntos de estructura: 200
Velocidad base: 5
Tiempo de fabricación base: 22 minutos.
Nivel de tecnologia minima: 5

Tiempo de fabricación

Los robots y vehículos requieren un tiempo para ser fabricados dependiente del tipo de unidad a fabricar, de la cantidad de territorios capturados por el jugador y del daño recibido por la estructura que fabricará a la unidad (léase Fábrica o Fuerte).

La siguiente ecuación resume lo anterior:

$$TiempoFabricacionFinal = \lfloor (TiempoFabricacionBase / TerritoriosTomados) / sqrt(1 - DañoEstructuraRel) \rfloor$$

Donde:
TiempoFabricacionBase es el tiempo de base propio de cada robot o vehículo. Por ejemplo los Tanques ligeros tienen 14 minutos (o 840 segundos) de tiempo de fabricación.
TerritoriosTomados es simplemente la cantidad de territorios que el jugador controla.
DañoEstructuraRel es un número entre 1.0 y 0.0 que mide cuánto es el daño que la estructura sufrió. Por ejemplo, un Fuerte consta de 1000 puntos de estructura y si recibió el impacto de 4 Misiles que cada uno hace 25 puntos de daño, el daño total es de 100 puntos y el relativo es de 100/1000 o 0.1.

Ejemplo:
El tiempo de fabricación final de un Tanque ligero por un Fuerte con 100 puntos de daños por parte de un jugador con 4 territorios bajo su control sería de:

$$TiempoFabricacionFinal = \lfloor (840 / 4) / sqrt(1 - (100/1000)) \rfloor = \lfloor 210 / sqrt(0.9) \rfloor = \lfloor 221.35943 \rfloor = 221$$

Velocidad de los robots

La velocidad de los robots está directamente relacionada con el tipo de terreno por donde quiere caminar (véase la sección Tipos de terrenos) y es independiente del tipo de robot (vease la seccion Robots).
Todos los robots tienen una velocidad base de 4 y la velocidad final está dada por:

$$VelocidadFinal = max(\lfloor 4 * FactorTerreno \rfloor, 1)$$

Donde:
FactorTerreno es 1.5 para los terrenos Camino/Carretera asfaltada, 0.7 para el Agua/Pantano y 1 para el resto de los terrenos (véase la sección Tipos de terrenos)

Nótese como la velocidad mínima es 1; ningún robot puede ir más lento que dicho valor.

Ejemplo:
Un grupo de robots de tipo Gruñido (Grunt) camina por el agua. Su velocidad es

$$VelocidadFinal = \lfloor 4 * 0.7 \rfloor = \lfloor 2.8 \rfloor = 2$$

Velocidad de los vehículos

La velocidad de los vehículos dependerá del tipo de vehículo, el daño recibido y el tipo de terreno por donde quiere transitar:

$$VelocidadFinal = max(\lfloor VelocidadVehiculoBase * FactorTerreno * (1 - DañoVehiculoRel) \rfloor, 1)$$

Donde:
VelocidadVehiculoBase es la velocidad intrínseca del vehículo. Por ejemplo un Tanque ligero tiene una velocidad base de 6.
FactorTerreno es 1.5 para los terrenos Camino/Carretera Asfaltada y 1 para el resto de los terrenos (véase la sección Tipos de terrenos)
DañoVehiculoRel es un número entre 1.0 y 0.0 que mide cuánto es el daño que el vehículo sufrió.

Nótese como la velocidad mínima es 1; ningún vehículo puede ir más lento que dicho valor.

Ejemplo
La velocidad de un Tanque ligero de velocidad base 6 y 80 puntos de estructura que recibió 50 puntos de daño sobre un Camino es de:

$$VelocidadFinal = max(\lfloor 6 * 1.5 * (1 - (50/80)) \rfloor, 1) = max(\lfloor 9 * (1 - 0.625) \rfloor, 1) = max(\lfloor 3.375 \rfloor, 1) = 3$$

Jugabilidad

Cada jugador debe poder ver una porción del mapa con una vista de tipo agila similar a la de otros juegos de estrategia.
Se debe poder distinguir claramente qué unidades le pertenecen a qué jugador. Se podrán utilizar imágenes de distintos colores, un color por jugador u otras variantes.

El jugador podrá seleccionar una unidad o varias unidades con el mouse y al hacer *click* sobre un terreno, estas unidades se desplazarán hacia dicha posición.
Si el *click* es sobre una unidad o estructura enemiga, las unidades se dirigirán hacia su objetivo y lo atacarán. En el caso de que el objetivo se mueva, las unidades que lo persiguen deberán reajustar sus trayectorias (deberán ir tras su objetivo).

Como otros juegos de estrategia el jugador solo indica el destino y es el motor del juego quien encuentra el camino más corto desde el punto de partida al destino para las unidades seleccionadas.
Para ello se deberá implementar el algoritmo **A-Star (o A*)**

Tanto los robots como los vehículos deben atacar automáticamente a las unidades enemigas sin que le

jugador lo indique si ocurren simultáneamente las siguientes condiciones:

- Las unidades enemigas están en el alcance de las unidades del jugador.
- Las unidades del jugador están quietas (el jugador no les dio una orden explícita de moverse o atacar)

En ningún caso las unidades del jugador deben moverse o perseguir al enemigo de forma autónoma sin una orden del jugador.

Mensajes de voz

Dado que el jugador no puede estar viendo todo el mapa a la vez, el juego debe de notificarle de los eventos más importantes que transcurren con mensajes de voz:

- Cuando un territorio es tomado por el jugador
- Cuando un territorio del jugador es perdido y tomado por un enemigo
- Cuando una unidad del jugador es creada
- Cuando una unidad del jugador es atacada
- Cuando una estructura que el jugador está atacando es destruida
- Cuando el Fuerte del jugador está bajo ataque.
- Cuando se hace *click* en una unidad del jugador se debe decir el tipo de unidad seleccionada.

Si la cantidad de eventos que suceden es muy grande, algunas mensajes de voz pueden ser evitados para no saturar al jugador con tanta información.

Sonidos

Como todo juego se debe reproducir sonidos para darle realismo a los eventos y acciones que suceden:

- Cuando hay disparos de misiles se debe emitir el sonido característico de un misil. Lo mismo aplica para el lanzallamas y el láser.
- Cuando hay una explosión se debe emitir un sonido acorde.

Si la cantidad de eventos que suceden es muy grande, algunos sonidos pueden ser evitados para no saturar al jugador con tanta información.

Animaciones

Tanto las unidades como los edificios no son mostrados con imágenes estáticas sino con pequeñas animaciones.
Los robots deben tener una animación mientras se desplazan por el terreno o atacan así como también los vehículos.

Interfaz del jugador

Se debe mostrar la parte del mapa que el jugador está viendo permitiéndole moverse al desplazar el mouse o al usar las teclas de dirección (arriba, abajo, izquierda, derecha). La vista que el jugador posee es conocida como *vista de águila*, en donde el jugador puede ver el mapa y sus elementos desde arriba.

Al seleccionar una unidad del jugador se debe poder ver una imagen de la cara del robot seleccionado (o del conductor en el caso de un vehículo) y el tipo de armamento que tiene así como también los puntos de daño recibidos.

Al seleccionar una Fábrica o Fuerte del jugador se debe poder ver qué unidades se pueden crear, el tiempo que llevaría y el tiempo que le resta terminar la unidad actual. Se debe poder seleccionar qué unidad se quiere construir. Una vez iniciado el proceso de construcción, se deben poder mostrar cuánto tiempo resta para terminar la unidad actual.

Screenshot de una posible interfaz gráfica mostrando: robots, vehículos, distintos tipos de terreno, una fábrica y una bandera, entre otros elementos. A la derecha se muestra a la unidad seleccionada. La interfaz final del trabajo puede variar respecto de esta muestra siempre que respete los requerimientos indicados.

Aplicaciones Requeridas

Servidor

El juego a implementar es multijugador con una arquitectura cliente-servidor. El servidor deberá recibir por parámetro la ruta a un archivo de configuración que contendrá:

- Puerto donde escuchar
- Rutas a los archivos que contienen los diferentes niveles o mapas.
- Parámetros del juego: el daño de cada unidad, las velocidades, todos los valores numéricos descritos en el presente enunciado deben ser configurables desde un archivo de texto. Esto es esencial para optimizar la jugabilidad y balancear las fortalezas y debilidades de las unidades.

Cliente

Es el elemento central que interactúa con el jugador. Debe poder mostrarle una pantalla de login para definir a qué servidor quiere conectarse, qué nivel o mapa desea jugar y con quienes formar un equipo.
Ya iniciado el juego, debe debe ofrecer una interfaz rica de acuerdo a los requerimientos pedidos y al

finalizar cada partida debe presentar las pantallas de victoria o derrota.

Generador de mapa

Los mapas pueden ser generados de forma automática a través de una aplicación adicional. Debe recibir por parámetro

- El tamaño del mapa.
- La cantidad de territorios.
- El o los niveles de tecnología de las fábricas que existirán, los tipos (de robots y de vehículos) y la cantidad.
- Porcentaje de territorios de agua/pantano y lava. Estos deben formar rios.
- Se debe garantizar que todos los fuertes y fábricas de vehículos tengan un camino (por tierra o por puentes) hacia el resto de los territorios. En otras palabras, un fuerte o fábricas de vehículos no puede estar en una isla, aislado del resto del mapa.
- Porcentaje de variedad de terrenos.
- Cantidad de vehículos abandonados.

Distribución de Tareas Propuesta

Con el objetivo de organizar el desarrollo de las tareas y distribuir la carga de trabajo, es necesario planificar las actividades y sus responsables durante la ejecución del proyecto. La siguiente tabla plantea una posible división de tareas de alto nivel que puede ser tomada como punto de partida para la planificación final del trabajo:

	Alumno 1 **Servidor - Modelo**	**Alumno 2** **Modelo - Cliente**	**Alumno 3** **Cliente - Generador**
Semana 1 (02/05/2017)	Implementación del algoritmo A-Star sobre un mapa con distintos terrenos y con unidades con diferentes movilidades. Un robot o vehículo debería poder encontrar el camino a su objetivo.	Mostrar una imagen. Mostrar una animación. Mostrar ambas en un lugar fijo o desplazándose por la pantalla (movimiento).	Emitir un sonido. Carga y guardado de información a disco. Draft de la interfaz (*wireframes*).
Semana 2 (09/05/2017)	Finalización del algoritmo de A-Star. Lógica del modelo sobre el ataque y destrucción de unidades y edificios.	Mostrar todo el mapa, incluyendo varios tipos de territorios distintos, puentes, fábricas, banderas, robots y vehículos.	Mostrar una interfaz para el jugador, excepto el mapa. Incluye la vista de las caras de los robots seleccionados y crear nuevas unidades (fábricas y fuertes): mostrar unidades disponibles, sus tiempos y la unidad actual a crear.
Semana 3 (16/05/2017)	Lógica del modelo sobre la creación de los robots y vehículos.	Movimiento de la cámara pero ninguna animación o explosión o disparo.	Generación de mapas simples: terrenos y algunos objetos con

			posiciones fijas.
Semana 4 (23/05/2017)	Lógica de modelo sobre captura de territorios y condiciones de victoria y derrota.	Selección de unidades y fábricas. Acciones de mover y atacar. Selección de qué unidad crear.	Creación de mapas complejos con todos los objetos sobre él, desde rocas hasta robots y fábricas. Carga de los mapas en el servidor.
Semana 5 (30/05/2017)	Creación de una partida multijugador de N jugadores. Formación de equipos.	Mostrar las unidades con animaciones. Mostrar las animaciones de los disparos y explosiones.	Pantalla de login con el servidor. Formación de equipos. Pantallas de derrota y victoria. Selección del nivel o mapa a jugar.
Semana 6 (06/06/2017)	- Pruebas y corrección sobre estabilidad del servidor. - Detalles finales y documentación preliminar	- Pruebas y correcciones en la jugabilidad del cliente. - Detalles finales y documentación preliminar	- Pruebas y correcciones en la performance del cliente. - Detalles finales y documentación preliminar
Preentrega el 13/06/2017			
Semana 7 (13/06/2017)	- Testing y corrección de bugs - Documentación	- Testing y corrección de bugs - Documentación	- Testing y corrección de bugs - Documentación
Semana 8 (20/06/2017)	- Correcciones sobre Preentrega - Testing - Documentación - Armado del entregable	- Correcciones sobre Preentrega - Testing - Documentación - Armado del entregable	- Correcciones sobre Preentrega - Testing - Documentación - Armado del entregable
Entrega el 27/06/2017			

Restricciones

La siguiente es una lista de restricciones técnicas exigidas por el cliente:

1. El sistema se debe realizar en ISO C++ utilizando librerías gtkmm, cairomm[2] y/o SDL[3].
2. Con el objetivo de minimizar tiempos y posibles errores, se permiten usar distintas librerías externas con la aprobación previa del tutor.
3. La información de mapas deben ser almacenados en formato XML o JSON. A tal fin, y con el objetivo de minimizar tiempos y posibles errores, se permiten distintas librerías externas (consultar sitio de la cátedra). No está permitido utilizar una implementación propia de lectura y escritura de XML o JSON.
4. Es condición necesaria para la aprobación del trabajo práctico la entrega de la documentación mínima exigida (consultar sitio de la cátedra). Es importante recordar que cualquier elemento faltante o de dudosa calidad pone en riesgo la aprobación del ejercicio.

Referencias

[1] Z de Bitmap Brothers: http://www.bitmap-brothers.co.uk/our-games/past/z.htm
[2] cairomm: http://www.cairographics.org/cairomm/
[3] SDL: https://www.libsdl.org