Fiuba
Facultad de Ingeniería
Universidad de Buenos Aires

U.B.A. FACULTAD DE INGENIERÍA

Departamento de Computación

Taller de Programación 75-42
Informática

TRABAJO PRÁCTICO FINAL
Z

Manual de Proyecto

Curso 2017 - 1er Cuatrimestre

GRUPO N° 2	
APELLIDO, Nombres	**N° PADRÓN**
MAITIA, Darius	95436
BOSCH, Martin	96749
TEJERINA, Luis	96629

Índice

Objetivo

El objetivo del trabajo práctico es desarrollar un remake[1] en C++ (y con la ayuda de diversas herramientas que se documentan más adelante) del juego Z [2] desarrollado en 1996 por The Bitmap Brothers.

División de tareas y evolución del proyecto

La división de tareas seguida por el grupo de trabajo fue a grandes rasgos aquella propuesta en el enunciado del trabajo práctico que se muestra en la tabla 1 siguiente; Alumno 1 (Servidor - Modelo) corresponde a Luis Tejerina, Alumno 2 (Modelo - Cliente) corresponde a Martín Bosh y Alumno 3 (Cliente - Generador) a Darius Maitia.

[1] El enunciado se adjunta en el anexo.
[2] "Z (video game) - Wikipedia." https://en.wikipedia.org/wiki/Z_(video_game).

	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			

Tabla 1: División de tareas

Semana 1

- Alumno 1:
    - Implementación del algoritmo A-Star
    - Modelo del mapa
- Alumno 2:
    - Mostrar una animación
    - Mostrar imagen desplazándose por la pantalla.
- Alumno 3:
    - Emitir un sonido.
    - Carga y guardado de información a disco.
    - Draft de la interfaz (wireframes).

Semana 2

- Alumno 1:
    - Lógica del modelo sobre ataque y destrucción.
- Alumno 2:
    - Imágenes del mapa.
    - Imágenes de robot Grunt.
- Alumno 3:
    - Mostrar una interfaz para el jugador, excepto el mapa.

Semana 3

- Alumno 1:
    - Lógica del modelo sobre creación de robots y vehículos.
- Alumno 2:
    - Animaciones de ataque, movimiento y quieto.
- Alumno 3:
    - Continuación de la interfaz del jugador (sin las caras de los jugadores ni las armas), menús, botones.
    - Generación de mapas simples: terrenos y algunos objetos con posiciones fijas.
    - Creación de mapas complejos con todos los objetos sobre él, desde rocas hasta robots y fábricas.
    - Almacenamiento y carga de los mapas en formato .json.

Semana 4

- Alumno 1:
    - Finalización de las lógicas del modelo: condiciones de captura y de victoria y derrota.
- Alumno 2:
    - Movimiento de la cámara.
- Alumno 3:
    - Carga de los mapas (cliente y servidor).
    - Refactoring sobre los menús y botones.

Semana 5

- Alumno 1:
    - Rocas y puentes en el modelo.
    - Serialización de eventos.
- Alumno 2:
    - Imágenes de los tipos de balas.
    - Animaciones de explosiones.
- Alumno 3:
    - Continuación de los menús y adaptación de estos con el código del proyecto

Semana 6

- Alumno 1:
    - Separación del proyecto en cliente-servidor con threads.
    - Lógica del inicio de partida
- Alumno 2:
    - Distintas animaciones de muertes y de unidad quieta
    - Refactoring de la vista.
- Alumno 3:
    - Interfaz del generador de mapas
    - Finalización del menú de producción

Semana 7

- Alumno 1:
    - Corrección de bugs del servidor
- Alumno 2:
    - Animaciones de los vehículos, con su top/arma separada del vehículo en sí.
    - Mejora en el movimiento de las unidades.
- Alumno 3:
    - Creación del side board.
    - Creación de pantallas de victoria y derrota.

Semana 8

- Alumno 1:
    - Documentación.
    - Corrección de bugs.
    - Refactoring del código.
- Alumno 2:
    - Mostrar edificios como animaciones, en vez de una imagen quieta.
    - Documentación.
    - Corrección de bugs.
    - Refactoring del código.
- Alumno 3:
    - Creación de interfaz de conexión del cliente.
    - Mostrar vida de unidad en el side board.
    - Documentación.
    - Corrección de bugs.
    - Refactoring del código.

Inconvenientes encontrados

Falta de experiencia y de asesoramiento a la hora de hacer un trabajo grande del estilo, con herramientas desconocidas como GTK, SDL2, JSon, Callgrind, Cmake.
Poco tiempo para cumplir adecuadamente con las entregas, particularmente al inicio.
Inexperiencia a la hora de usar un controlador de versiones.
Dificultad al separar el proyecto en cliente y servidor.
Dificultad a la hora de acoplar las distintas funcionalidades hechas por cada uno de los integrantes.

Análisis de puntos pendientes

- Agregar sonidos al juego.
- Mejorar lógica de movimiento de unidades.
- Mejorar algunas animaciones, por ej:
    - Al destruir un edificio.
    - impacto de bala.
    - top/arma del jeep, que suba y baje con el vehículo.
- Mejores imágenes de los distintos terrenos, rocas y puente.
- Agregar animaciones en el menu de producción y en el side board.
- Eficiencia en el server, por ej, el lag que ocurre entre indicarle a una que se mueva y esta se empiece a mover.

Herramientas

- **CLion**
- **CMake**
- **GitHub**
- **Gimp**
- **Glade**
- **GDB, Callgrind, Valgrind, KCacheGrind**
- **Google-Tests**
- **Google docs**
- **Draw.io**