Argentum

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 13 (07/07/2020). **Entrega:** clase 15 (21/07/2020).
Temas de Repaso	● Aplicaciones Cliente-Servidor multi-threading. ● Interfaces gráficas ● 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

El presente trabajo consistirá en recrear el mítico juego Argentum On-line [1], un juego multijugador en donde los participantes controlan a un personaje de rol en un mundo fantástico de magia y lleno de criaturas salvajes.

Descripción

Tierras de Argentum

Argentum es un lugar extenso rico y variado desde zonas boscosas hasta desiertos.

El tipo de terreno sin embargo no afecta la jugabilidad en ningún modo (da lo mismo caminar sobre un camino que sobre arena).

Por supuesto, en las zonas boscosas hay árboles.

Esparcidas sobre Argentum hay edificaciones y construcciones (que no están dentro de las ciudades ni pueblos) que no tienen ninguna utilidad particular salvo la de obstaculizar el paso. Como es de suponerse, los personajes no pueden atravesar las paredes!

También es cierto que los personajes no pueden atravesarse entre sí ni con los NPC: las personas colisionan, incluso los fantasmas!

Puntos de vida

Son los puntos que determinan la salud del personaje. Si llegan a 0 el personaje muere y se transforma en fantasma.

Los puntos de vida máximos de un personajes están dados por la siguiente ecuación:

```
VidaMax = Constitución * FClaseVida * FRazaVida * Nivel
```

Los puntos de vida pueden recuperarse (hasta llegar al máximo) tomando alguna poción curativa o con algún hechizo de curación.

También pueden recuperarse con el paso del tiempo según esta ecuación:

```
Vida = FRazaRecuperacion * segundos
```

Puntos de mana

Todos los personajes salvo el Guerrero pueden realizar hechizos de magia. Cada hechizo requiere un mínimo de maná para realizarse y consume la misma cantidad.

Los puntos de maná máximos están dados por:

```
ManaMax = Inteligencia * FClaseMana * FRazaMana * Nivel
```

Los puntos de maná pueden recuperarse (hasta llegar al máximo) tomando alguna poción mana o realizando meditación o simplemente por el paso del tiempo.

Un personaje puede meditar con el comando **`/meditar`**. En el estado de meditación el jugador no puede realizar otra acción, cualquier acción lo sacaran del estado de meditación.

La recuperación de maná está dada por:

```
Mana = FClaseMeditacion * Inteligencia * segundos
```

La recuperación por el mero paso del tiempo está dada por:

```
Mana = FRazaRecuperacion * segundos
```

Oro

En Argentum la moneda de comercio es el oro. Un personaje puede comprar y vender objetos intercambiando oro.

Además, cada vez que el jugador mate que un NPC habrá cierta cantidad de oro que el NPC "dejará caer" (llamado *drop*).

Este está dado por la siguiente ecuación:

```
Oro = rand(0, 0.2) * VidaMaxNPC
```

Si el jugador mata a otro jugador, el oro obtenido será exactamente el oro que el jugador muerto tenía "en exceso".

Cada personaje tiene un máximo de oro que puede tener en mano "seguro"; el personaje podrá tener hasta un adicional del 50% y este oro se lo considera "en exceso".

Por ejemplo si un personaje tiene como máximo 1000 de oro podrá tener en mano hasta 1500. Si el personaje cuenta a la hora de morir con 900 de oro, nada de su oro se caerá al suelo (el muerto no lo perderá) por ser 900 < 1000 (su máximo).

Si en cambio tenía 1100, el muerto conservará 1000 y dejara caer al suelo 100 de oro.

El máximo de oro en mano "seguro" que cada jugador tiene esta dado por:

```
OroMax = 100 * Nivel^1.1
```

Inventario

Cada jugador podrá tener hasta un máximo de **`N`** objetos en su inventario. Estos incluyen armas, armaduras, cascos, escudos, pociones, báculos (hechizos) entre otros.

Algunos de estos objetos pueden ser equipados, es decir, ser usados por el jugador.

Por ejemplo, cuando un jugador tiene un arma equipada, el daño que produzca en un ataque estará determinado por esta.

Un jugador puede tener múltiples armas, armaduras, cascos, escudos y varas/báculos (hechizos) pero sólo podrá tener equipado uno de cada uno.

No se podrá tener un arma y una vara/báculo equipados a la vez.

Otros objetos, como las pociones, cuando son equipadas son consumidas por el personaje teniendo un efecto inmediato.

El jugador podrá recolectar objetos del suelo posicionándose sobre ellos y ejecutando el comando **`/tomar`**.

De forma inversa, el jugador podrá seleccionar un objeto de su inventario y dejarlo caer con **`/tirar.`**

Puntos de experiencia

Cada personaje tendrá puntos de experiencia que, superados un límite, harán que el jugador pase al siguiente nivel.

El límite para alcanzar el siguiente nivel está dado por:

```
Limite = 1000 * Nivel^1.8
```

La experiencia se consigue atacando a un NPC o a otro jugador.

Por cada ataque realizado se obtiene este cantidad de experiencia:

```
Exp = Daño * max(NivelDelOtro - Nivel + 10, 0)
```

donde **`NivelDelOtro`** es el nivel del NPC o del jugador que está siendo atacado.

Al matar a un NPC o a otro jugador se obtendrá una experiencia adicional:

```
Exp = rand(0, 0.1) * VidaMaxDelOtro * max(NivelDelOtro - Nivel + 10, 0)
```

donde **`VidaMaxDelOtro`** es la vida máxima del NPC o del otro jugador.

Ataque

Un personaje puede atacar a un NPC o a otro jugador restándole puntos de vida.

El daño producido esta dado por:

```
Daño = Fuerza * rand(DañoArmaMin, DañoArmaMax)
```

Existe cierta probabilidad de realizar un ataque crítico que causa el doble de daño y no puede ser esquivado.

Si el arma equipada es una arma con rango, por ejemplo un arco, o bien es un hechizo, el jugador puede realizar el ataque a distancia haciendo click sobre el NPC o jugador.

En otro caso, el ataque se realiza solo si el jugador está al lado del adversario (jugador o NPC) que quiera atacar.

Defensa

Ante un ataque un personaje o NPC puede defenderse.

Primero, existe la posibilidad de esquivar el ataque y recibir un daño nulo:

```
Esquivar si rand(0, 1) ^ Agilidad < 0.001
```

Si el ataque no pudo ser esquivado le personaje recibirá un daño pero este podrá ser reducido según la defensa que le provea su armadura, escudo y/o casco:

```
Defensa = rand(ArmaduraMin, ArmaduraMax) + rand(EscudoMin, EscudoMax) +
rand(CascoMin, CascoMax)
```

Si un jugador recibe un daño de 10, pero tiene una defensa de 8, solo se le restaran 2 puntos de vida.

Razas y Clases

Al momento de crear el personaje se podría elegir la raza y la clase del personaje.

Cada raza y cada clase afectan de una u otra manera a las ecuaciones que gobiernan el juego.

Por ejemplo, **`FRazaRecuperación`** afecta como un personaje recupera sus puntos de vida y maná.

Hay 4 razas:
- **Humanos**: son una raza equilibrada.
- **Elfos**: son muy inteligentes y ágiles pero de una constitución física frágil
- **Enanos**: son muy fuertes y resistentes, pero la agilidad no es lo suyo
- **Gnomos**: inteligentes y resistentes, pero mucho menos ágiles que los elfos.

Hay 4 clases:
- **Mago**: quienes hayan estudiado magia han cultivados sus mentes y menos sus cuerpos
- **Clérigo**: un poco menos inteligentes y hábiles que los magos, un clérigo compensa sus falencias con un mayor desempeño físico.
- **Paladin**: entrenados para el combate, son fuertes y resistentes aunque capaces también de usar magia, con una mucho menor inteligencia que un mago.
- **Guerrero**: han dedicado toda su vida al combate, son típicamente más fuertes y resistentes que otros pero carecen de la sabiduría para usar la magia.

De todas las clases, el guerrero es el único que no puede usar la magia, ni meditar y su mana es siempre 0.

Muerte

Cuando un jugador muere este puede perder oro (véase la sección Oro) y/o experiencia (véase la sección Experiencia).

Ademas, todos los objetos de su inventario caen al piso.

Tras la muerte, el personaje entonces se convierte en fantasma.

Un fantasma puede moverse como cualquier otro jugador solo que no podrá interactuar con nadie ni con nada.

El jugador podrá dirigirse al sanador de la ciudad más próxima para resucitar o bien podrá ingresar el comando **`/resucitar`**.

En este último caso, el jugador aparecerá resucitado junto al sanador de la ciudad más próxima. Esta operación no es inmediata y el fantasma estará inmovilizado un tiempo proporcional a la distancia entre él y el sanador antes de trasladarse hacia él y resucitar.

Fair play

Un jugador con un nivel 12 o menor es considerado "*newbie*". Los newbies no pueden atacar ni ser atacados por otros jugadores.

Un jugador no podrá atacar ni ser atacado por otro jugador si la diferencia de niveles entre ellos es superior a 10.

Ciudades y pueblos

A lo largo de las tierras de Argentum hay pequeñas ciudades y pueblos en las que los jugadores podrán encontrar al menos a un sacerdote, un comerciante y un banquero.

Los jugadores pueden interactuar con ellos seleccionándolos (click) y escribiendo algún comando:

Sacerdote: puede resucitar a un personaje (**`/resucitar`**), puede curar a un personaje herido tanto en vida como en mana (**`/curar`**) y puede vender báculos y varas (hechizos) y pociones (**`/comprar <objeto>`**)

Comerciante: puede comprar y vender casi cualquier artículo incluyendo armas, armaduras y cascos. Algunos compran y venden pociones pero nunca hechizos (**`/comprar <objeto>`** y **`/vender <objeto>`**)

Banquero: los jugadores pueden guardar sus pertenencias en el banco incluyendo el oro (**`/depositar <objeto>`**) y por supuesto pueden retirarlos del banco (**`/retirar <objeto>`**) En el caso del oro podrá

especificar el monto: **`/depositar oro <cant>`** y **`/retirar oro <cant>`**

El sistema bancario de Argentum es sin duda muy eficiente: cuando un jugador hace un depósito, el jugador puede hacer un retiro desde cualquier sucursal en todo Argentum sin cargo.

Las ciudades y pueblos son zonas seguras: los jugadores no pueden atacar ni ser atacados ni los NPC pueden entrar.

Items

Existen los siguientes items:

- **Espada**: arma de combate a mano, tiene un daño de 2 a 5.
- **Hacha**: arma de combate a mano, tiene un daño de 4 a 5.
- **Martillo**: arma de combate a mano, tiene un daño de 1 a 9.
- **Vara de fresno**: permite lanzar el hechizo "*flecha mágica*" que tiene un daño a distancia de 2 a 4. Consume 5 de mana.
- **Flauta élfica**: permite lanzar el hechizo "*curar*" sobre el jugador que restaura la vida. Consume 100 de mana.
- **Báculo nudoso**: permite lanzar el hechizo "*misil*" que tiene un daño a distancia de 4 a 8. Consume 15 de mana.
- **Báculo engarzado**: permite lanzar el hechizo "*explosion*" que tiene un daño a distancia de 8 a 20. Consume 30 de mana.
- **Arco simple**: arma a distancia, tiene un daño de 1 a 4. Las flechas son infinitas.
- **Arco compuesto**: arma a distancia, tiene un daño de 4 a 16. Las flechas son infinitas.
- **Armadura de cuero**: ofrece una defensa entre 2 y 6
- **Armadura de placas**: ofrece una defensa entre 15 y 30
- **Tunica azul**: ofrece una defensa entre 6 y 10
- **Capucha**: casco que ofrece defensa entre 1 y 4
- **Casco de hierro**: ofrece defensa entre 4 y 8
- **Escudo de tortuga**: ofrece defensa entre 1 y 2
- **Escudo de hierro**: ofrece defensa entre 1 y 4
- **Sombrero mágico**: casco que ofrece una defensa entre 4 y 12.

Criaturas o NPC

Hay varias criaturas en Argentum listas para atacar a un jugador si este se encuentra lo suficientemente cerca, incluso si el jugador no inicia el ataque.

Todas las criaturas tiene un ataque cuerpo a cuerpo y se dirigirán al jugador más cercano para atacarle (si es que está lo suficientemente cerca)

Cada cierto tiempo un cierto número de criaturas serán creadas hasta que la población de estas llegue a un límite, lo que garantiza que en Argentum haya siempre criaturas sin llegar a saturar las tierras.

Con la muerte de cada criatura existe cierta probabilidad de que esta deje algún objeto u oro:

```
Probabilidad -> Item dejado
0.80          Nada
0.08          Oro, una cantidad igual a rand(0.01, 0.2) * VidaMaxNPC
0.01          Un poción de vida o mana elegida al azar
0.01          Cualquier otro objeto al azar
```

Las criaturas existentes son:

- **Goblin**
- **Esqueleto**
- **Zombie**
- **Araña**

Interfaz de usuario

El juego muestra al personaje desde una vista de águila con el personaje en el centro de la pantalla.

El jugador puede moverse usando el teclado y puede atacar haciendo click sobre otro jugador o NPC.

La interfaz debe mostrar también el inventario, cuales objetos están equipados, la cantidad de oro, vida y mana del jugador y su experiencia y nivel.

La imagen esta a efectos ilustrativos. No se requiere que la UI sea igual a esta. La imagen pertenece a una variante del juego original conocida como OptimizAO.

Mini-chat

La interfaz cuenta con un mini-chat que muestra los últimos mensajes relevantes para el jugador y un text-input para que el jugador pueda escribir.

Los mensajes relevantes son:
- Daño provocado
- Daño recibido
- Evasion (si el jugador pudo esquivar un ataque o el contrincante lo hizo)
- Mensajes de otro jugador destinados a él

El jugador puede escribir los siguientes mensajes. Algunos requieren que el jugador seleccione a otro personaje para dirigirle el mensaje.

- **`/meditar`**
  El jugador entra en estado de meditación y recupera mana (ver Puntos de mana).
- **`/resucitar`**
  Si el jugador es un fantasma, se traslada a la ciudad más cercana y resucita. Sino, el mensaje debe ser enviado a un sacerdote (ver Muerte).
- **`/curar`**
  Recupera los puntos de vida y maná. El mensaje lo debe recibir un sacerdote.
- **`/depositar <objeto>`**
  Toma el objeto del inventario del jugador y lo pone en el banco. El mensaje debe ser enviado a un banquero.
- **`/retirar <objeto>`**
  Recupera un objeto del banco y lo guarda en el inventario. El mensaje debe ser enviado a un banquero.
- **`/listar`**
  Lista los objetos que el comerciante tiene para vender o los que el banquero tiene.
- **`/comprar <objeto>`**
  Comprar un objeto. El mensaje debe ser enviado a un sacerdote o comerciante.
- **`/vender <objeto>`**
  Vende un objeto. El mensaje debe ser enviado a un comerciante.
- **`/tomar`**
  Recoge del suelo un objeto y lo guarda en el inventario.
- **`/tirar`**
  Tira el objeto del inventario seleccionado.
- **`@<nick> <msj>`**
  Le envía un mensaje privado a otro jugador.

Vestimenta

Todo personaje tendrá una vestimenta que irá cambiando según la armadura, casco, sombrero, arma y/o báculo equipado y se mostrará así en pantalla.

Si no se tiene una armadura equipada, se debe mostrar una vestimenta común.

Persistencia

Se debe garantizar que los avances de los jugadores sean persistidos. Así cuando un jugador vuelva a jugar podrá continuar con su personaje en el mismo lugar en donde estaba al momento de salir.

Es importante que la persistencia se haga periódicamente en caso que el jugador tenga una desconexión abrupta.

Se recomienda usar 2 archivos:

- El primero contiene los datos de cada jugador donde estos son structs de tamaño fijo y constante.
- El segundo es un diccionario (map) que mapea el nombre de un jugador con la posición (offset) de los datos del jugador en el primer archivo.

No se puede cargar todo el archivo a memoria (se presume que habrá miles de jugadores pero que solo

unos cientos estarán online a la vez por lo que el costo de tener todo el archivo a memoria con todos los datos de todos los jugadores es un exceso.

Sin embargo se permite tener en memoria el segundo archivo que funciona como índice y es mucho más chico.

Los archivos deben ser binarios y el borrado de usuarios **no** será requerido.

Configuración

Es importante que cada ecuación esté programada en su propio método o función de C++ y que todas las ecuaciones se encuentren en una única clase o módulo.

Además, todos los valores numéricos (constantes numéricas) deben venir de un archivo de configuración de texto.

Tener las ecuaciones en un solo lugar les permitirán hacer modificaciones a la lógica del juego si alguna ecuación está desbalanceada y hace que el juego no sea divertido.

Un archivo de configuración le permitirá a ustedes hacer "*ajustes*" más finos sin recompilar y al docente le permitirá cambiar ciertos valores para facilitar la corrección (por ejemplo se puede poner vida infinita)

Sonidos

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

- Cuando hay un golpe de espada se debe emitir el sonido característico.
- Cuando hay una explosión se debe emitir un sonido acorde.

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

Musica

Se debe reproducir una música de fondo.

Animaciones

Las unidades no son mostradas con imágenes estáticas sino con pequeñas animaciones: debe mostrarse cómo se mueven con animaciones.

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

- 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 sin tener que recompilar el código.

La definición del protocolo de comunicación entre el cliente y el servidor queda a libre elección. Se requiere sin embargo que sea binaria (no puede ser texto).

Cliente

Es el elemento central que interactúa con el jugador. Debe poder mostrarle una pantalla de login para definir a qué servidor quiere conectarse.
Ya iniciado el juego, debe ofrecer una interfaz rica de acuerdo a los requerimientos pedidos.

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 (*las fechas marcan el día en que la seman se completa y se deberían tener los temas terminados*):

	Alumno 1 **Servidor - Modelo**	**Alumno 2** **Modelo - Cliente**	**Alumno 3** **Modelo - Cliente**
Semana 1 (09/06/2020)	Carga de mapas. Lógica de movimiento de los personajes y NPC.	Mostrar una imagen. Mostrar una animación. Mostrar ambas en un lugar fijo o desplazándose por la pantalla (movimiento).	Reproducir sonidos, musica. Mostrar texto en pantalla.
Semana 2 (16/06/2020)	Lógica de ataque, defensa, drops tanto de jugadores como de NPCs) Razas y clases.	Mostrar todo el mapa, incluyendo varios tipos de terrenos distintos, edificios y unidades (jugadores y NPC).	Mostrar la interfaz gráfica (barra de experiencia, items, vida)
Semana 3 (23/06/2020)	Lógica de las ciudades, “fair play”. Compra, venta y otros comandos. Experiencia y niveles.	Mostrar los objetos (drop). Interacción por parte del usuario.	Mini-chat tanto para leer los mensajes como para escribirlos. Mensajes dirigidos a un NPC y a un jugador.
Semana 4 (30/06/2020)	Sistema de comunicación (cliente - servidor) Servidor completo.	Cliente completo.	Pantalla de login. Persistencia y configuración.
Semana 5 (07/07/2020)	- Pruebas y corrección sobre estabilidad del servidor.	- Pruebas y corrección sobre estabilidad del cliente.	- Pruebas y corrección sobre estabilidad del cliente.

	- Detalles finales y documentación preliminar	- Detalles finales y documentación preliminar	- Detalles finales y documentación preliminar
Primera Entrega el dia 07/07/2020			
Semana 6 (14/07/2020)	- Correcciones sobre Primera entrega - Testing y corrección de bugs - Documentación	- Correcciones sobre Primera entrega - Testing y corrección de bugs - Documentación	- Correcciones sobre Primera entrega - Testing y corrección de bugs - Documentación
Semana 7 (21/07/2020)	- Testing - Documentación - Armado del entregable	- Testing - Documentación - Armado del entregable	- Testing - Documentación - Armado del entregable
Entrega Final el dia 21/07/2020			

Restricciones

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

1. El sistema se debe realizar en ISO C++ utilizando librerías *SDL, Qt*, *gtkmm u OpenGL*.
2. Con el objetivo de facilitar el desarrollo de las interfaces de usuario, se permite el uso de *Glade* u otro editor de interfaces gráficas.
3. Se puede utilizar un editor de mapas externo y cargarlos desde el servidor como *Tiled* [2]
4. Se debe usar una librería de parsing como *jsoncpp* [3] o similar.
5. Para la serialización binaria se puede usar una librería externa como *msgpack* [4] o similar.
6. 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.
7. De forma opcional, se sugiere la utilización de alguna librería del estilo xUnit. Si bien existen varias librerías disponibles en lenguaje C++, se recomienda optar por CxxTest [5] o CppUnit [6]
8. Todo socket utilizado en este TP debe ser bloqueante (es el comportamiento por defecto).

Referencias

[1] https://wiki.argentumonline.org/
[2] https://www.mapeditor.org/
[3] https://github.com/Kurento/jsoncpp
[4] https://msgpack.org/
[5] https://cxxtest.com/
[6] https://en.wikipedia.org/wiki/CppUnit