Implementación de un motor básico para la creación de 2D MMORPGs

Reinaldo Alejandro Jerez Contreras (Estudiante)
Daniel Antonio Moreno Córdova (Profesor guía)
Carrera de Ingeniería Civil en Computación
Universidad de Talca

20 de agosto de 2018

El presente documento constituye la presentación de la propuesta para el proyecto de titulación de la carrera de ingeniería civil en computación. Este proyecto consiste en la implementación de un motor básico para la creación de 2D MMORPGs; el cual a través del uso de nuevas tecnologías, apunte a mejorar aspectos de seguridad, velocidad y estabilidad respecto de motores actuales.

1. Descripción de la propuesta

La propuesta consiste en la implementación de un motor básico para la creación de 2D MMORPGs, utilizando tecnologías nuevas y con mayores prestaciones en cuanto a rendimiento, seguridad, portabilidad, entre otros; con el fin de entregar una base solida, que sea candidata a reemplazar los motores de hace una década y que aun se utilizan por las comunidades aficionadas a la practica de creación de 2D MMORPGs.

1.1. Conceptos básicos del proyecto

- Motor: *(En el contexto del proyecto)* Es un software que entrega un conjunto de herramientas genéricas para la creación de otros software de manera simplificada. [1]
- Gráficos 2D: Imágenes, modelos, etc. Que se encuentras representados en un espacio de dos dimensiones. Usualmente identificados por los ejes X e Y.[2]
- MMORPG: *(Massive Multiplayer Online Role Play Game)* Son un subgrupo de juegos enfocados en el genero de rol (Donde cada jugador es representado por un personaje en el juego), que están pensando para la interacción de cientos o miles de jugadores en un mismo "universo".[3]
- Motor 2D MMORPGs: Es una unión de los 3 términos anteriores. Consiste en un motor para desarrollar juegos de multijugador masivo en linea, utilizando gráficos en dos dimensiones.
- Renderización: Es el proceso de generar una imagen a partir de un modelo o datos establecidos.[4]
- Juego de rol: Son juegos donde los participantes representan un personaje en el mundo virtual del juego. Es labor de los participantes decidir en cierto nivel como se desenvolverá su personaje en este mundo virtual.
- Arquitectura Cliente-Servidor: Es una estructura de aplicaciones distribuida y necesaria para la implementación de un MMORPG. Consiste en un software que actúa como servidor, al cual se comunican todos los clientes que desean acceder. El servidor procesa los datos y luego comunica los datos relevantes a cada cliente.[5]
- Scrum: Es una metodología de desarrollo que se enfoca en ciclos de desarrollo llamados sprints. Al iniciar los ciclos se define el/los objetivos del sprint,

durante el desarrollo se revisan todos los aspectos del proyecto. Al finalizar un ciclo se hace un análisis retrospectivo del desempeño durante el ciclo.[6]

1.2. Contexto del proyecto

En la actualidad existen muchos motores para el desarrollo de juegos; un motor es básicamente un software que entrega distintas herramientas para facilitar el desarrollo de juegos u otros productos relacionados. Muchos de estos motores están orientados a grandes empresas con grandes equipos de trabajo, mientras que otros están enfocados en el desarrollo individual o para grupos reducidos.

Existe un subgrupo de estos últimos que se enfocan en personas con bajo nivel técnico, o que desean enfocar sus esfuerzos en generar contenido, por sobre la necesidad de implementar la lógica de los juegos. Este subgrupo corresponde en su mayor parte a motores orientados al desarrollo de videojuegos de rol.

Existen variadas alternativas de motores para la creación de videojuegos de rol, uno de los mas conocido es RPG Maker[7]. La primera versión de RPG Maker fue lanzada en 1995 y la ultima versión durante el 2015. RPG Maker se centra en la creación de videojuegos de rol de una solo jugador.

Tomando las ideas de RPG Maker y otros motores tantos motores de videojuegos de rol; se comenzaron a desarrollar una serie de motores para crear 2D MMORPGs.

Hasta el día de hoy se siguen desarrollando y utilizando estos motores, incluso proyectos de Kickstarter[8] han sido iniciados utilizando estos motores. Lamentablemente estos motores están construidos sobre una base antigua y en general con muchas deficiencias.

1.3. Definición del problema

La mayoría de los motores para crear 2D MMORPGs que existen en la actualidad, están desarrollados utilizando como base tecnologías antiguas (Por ejemplo Visual Basic 6, DirectX 8, Winsocks y Winforms) [9], y a pesar que muchas personas se dedican a seguir mejorándolos; es muy difícil escapar de las limitaciones que existen al trabajar con tecnologías prácticamente obsoletas. Las falencias mas destacables de estos motores son:

- **Rendimiento**: Los motores actuales soportan una cantidad bastante reducida de jugadores simultáneos, generalmente cantidades menores a 100 jugadores.

- **Seguridad**: Los motores actuales presentan arquitecturas que permiten al cliente tomar decisiones sobre ciertos aspectos, esta característica del diseño permite que sea fácil manipular datos desde el lado del cliente para usuarios experimentados.
- **Estabilidad**: La mayoría de los motor existentes presentas fallos en tiempo de ejecución, lo cual conlleva a caídas abruptas de los servicios. Esto se debe principalmente a todos los cambios y actualizaciones que han sufrido a lo largo de los años.
- **Portabilidad**: La mayoría de los motores existentes están restringidos a ser utilizados en Windows, incluso algunos están restringidos a ciertas resoluciones de pantalla.

1.4. Trabajo relacionado

Durante los años han surgido bastantes proyectos que intentan realizar mejoras a los motores existentes o que intenta implementar nuevos motores desde cero. La mayoría de estos proyectos son abandonados debido a la falta de tiempo de los desarrolladores (Basta con realizar una búsqueda en GitHub para encontrar proyectos de esta índole). El resto de ellos corresponden a mejoras sobre los motores existentes y que inevitablemente mantienen alguno los problemas mencionados anteriormente.

Tenemos el caso particular de Eclipse Origins el cual se encaja en ambos de los puntos mencionados anteriormente. Se han sacado distintas versiones de este, incluso algunas cambiando componentes completos, pero se mantiene con la misma base de hace una década. Por otra parte también se anuncio una versión "web.ªctualizada, la cual fue posteriormente abandonada.[9]

A continuación se exponen algunos de los proyectos mas relevantes.

- Eclipse Origins: Es uno de los motores para crear 2D MMORPGs mas famosos que ha existido. Tiene distintas versiones actualmente y muchos proyectos derivados de este. Su ultima versión planeaba ser una plataforma web, pero fue posteriormente cancelado su desarrollo.[10]
- Intersect Engine[11]: Es un proyecto que intenta mejorar las funcionalidades de los existentes motores basados en Visual Basic 6. La primera versión fue lanzada en 2016 y es actualmente usado por una solida comunidad. A pesar de que se ha actualizado el motor gráfico y el lenguaje, aun cuenta con problemas respecto a la portabilidad. Sin mencionar que es un proyecto de código cerrado. Durante mediados del año 2018 se volvió a impulsar su desarrollo.

- MystaliaEngine[12]: Es un proyecto vigente que busca implementar un motor de desarrollo para 2D MMORPGs basado en Javascript. Aun está en sus etapas iniciales y solo tiene implementado un sistema de mapas.

- Dot World Maker[13]: Es un motor multiplataforma implementando utilizando Javascript tanto en cliente como en servidor. El proyecto apuntaba a mejorar y resolver los problemas de los motores antiguos. A pesar de presentar mejoras substanciales, tenia muchos problemas de rendimiento, además de sus interfaces complejas y su sistema de pago donde el motor era prestado como servicio. Su desarrollo se detuvo durante el año 2017 pero aun se encuentra disponible para ser utilizado.

- MMORPG Engine de LazyPlanet[14]: Fue proyecto para realizar un motor de juego basado en C-Sharp, el cual reemplazara los antiguos motores basados en Visual Basic. El proyecto fue abandonado el año 2015.

1.5. Propuesta de solución

Dado que existe una amplia comunidad que utiliza y mejora estos motores para la creación de 2D MMORPGs y debido a los tantos problemas de los motores existentes, se plantea el crear un nuevo motor de creación de 2D MMORPGs, el cual apunte a subsanar los problemas de los motores existentes en la actualidad descritos en el punto anterior, todo esto a partir del uso de tecnologías actuales y el uso de arquitecturas adecuadas.

A través de esta propuesta se busca generar un motor con las características básicas, el cual se presente a la comunidad de desarrolladores como una alternativa a las herramientas actuales, en su mayoría obsoletas, y que eventualmente se traduzca en una continuación o expansión del mismo.

La propuesta contempla la implementación del motor, el cual consiste en un servidor, un cliente y un editor. *Las características especificas se presentan en la sección de Alcances.*

2. Objetivos

A continuación se expone el objetivo del proyecto y conjunto a el los objetivos específicos que nos permitan lograr este proyecto. El lograr parte del objetivo del proyecto depende de las decisiones tomadas durante el desarrollo y puede que no estén explícitamente en los objetivos específicos.

Objetivo general

- Implementar un motor básico para la creación de juegos 2D MMORPGs, utilizando tecnologías actuales y las arquitecturas adecuadas, apuntando a reparar las falencias de los motores existentes.

Objetivos específicos

- Implementar un servidor estable, rápido y seguro, el cual sea posible portar a distintas plataformas. *(Las métricas de estabilidad, rapidez, seguridad, simplicidad y portabilidad están detalladas en la sección de Metodología de Evaluación)*
- Implementar un cliente que sea rápido y pueda ser utilizado desde distintas plataformas.
- Implementar un editor que pueda ser utilizado desde distintas plataformas y permita a los usuarios crear el juego de manera ”simple”.
- Definir detalladamente los requisitos del proyecto, con énfasis en lo que se necesita y en lo que se espera lograr.
- Definir un diseño modular el cual sea coherente a los requisitos establecidos y con alto nivel de modularidad con el fin de mejorar la escalabilidad y la mantenibilidad.
- Utilizar una metodología ágil con el fin de afrontar las áreas mas peligrosas y desconocidas del desarrollo en si de una manera controlada.
- Realizar una evaluación del motor a través de experimentos que apunten a medir el cumplimiento de los objetivos y la experiencia de los usuarios al utilizarlo.
- Escribir una documentación del uso del motor, que abarque todos los puntos principales tanto para la creación como para la puesta en marcha de un juego.

3. Alcances

- Se desarrollará un motor básico para la creación de 2D MMORPGs, esto significa que solo contendrá funcionalidades acotadas respecto a un motor completo, dado que se espera que se continúe su crecimiento posterior al termino del proyecto. Las funcionalidades a cubrir son las siguientes:
  - Edición y visualización de mapas basados en tiles (Cliente-Servidor-Editor)
  - Administración de clases (Cliente-Servidor-Editor)
  - Creación de cuentas y autenticación de usuarios (Cliente-Servidor)
  - Administración de personajes (Cliente)
  - Configuraciones generales (Cliente-Servidor-Editor)
  - Manejo de recursos (Editor)
  - Chat global (Cliente-Servidor)
  - Comandos de administración (Servidor)
  - **Stretch goal**: Audio (Cliente)
- El motor contempla la implementación de un servidor, un cliente y un editor.
- Se utilizará una arquitectura cliente-servidor.
- No se desarrollará documentación extensiva, solo la básica para poder utilizar el motor. (Documentación entre 10 y 20 paginas que abarque la instalación, creación y publicación de un juego de ejemplo).
- El foco del desarrollo será para su uso en Windows, pero la elección de tecnologías e implementación de ellas, estará enfocada en permitir la portabilidad a otras plataformas. *(Esto se debe que a pesar de usar tecnologías multiplataforma, hay unas o varias configuración que varían según la plataforma en que se fuese a utilizar. Esto también significa una carga mayor en cuanto a tiempo de desarrollo total.)*
- Las tecnologías propuestas son: C-Sharp y SQL Server para el servidor. ElectronJS y Angular para el editor, y por ultimo WebGL y Angular para el cliente.

4. Metodología

Para la implementación del proyecto se utilizará la metodología ágil Scrum, adaptada para el desarrollo individual.

Scrum es un modelo de metodología de desarrollo ágil el cual define un conjunto de actividades y roles para el desarrollo de proyectos (Generalmente para grupos de desarrollo). En Scrum el trabajo se desarrolla en varios ciclos de trabajo, donde los ciclos de trabajo apuntan a desarrollar una funcionalidad la cual pueda ser rápidamente probada. Durante los ciclos de trabajo de Scrum se puede trabajar en cualquier parte del desarrollo ya sea requisitos, diseño, implementación, pruebas, etc. Cabe destacar que las actividades de Scrum son parte relevante dela metodología, ya que apuntan al mejoramiento continuo de los ciclos de desarrollo.[5]

Se decide por el uso de Scrum ya que algunas partes del desarrollo corresponden a temas complejos de implementar y por ende susceptibles a fallos. Scrum nos permite minimizar el posible impacto que estos fallos puedan tener al proveernos de una forma rápida de probar el avance de cada ciclo de trabajo. Además Scrum permite aprovechar de mejor manera el tiempo de trabajo, ya que se puede trabajar directamente en lo que es necesario y no es necesario pasar por el ciclo estricto de requisitos, diseño, implementación.

Para la organización de las tareas por realizar, se utilizará un tablero online. En este caso se utilizará la herramienta Trello[15]. Además, es importante destacar que dado que este es un proyecto que será implementado por un solo desarrollador; las actividades de Scrum serán adaptadas de la siguiente manera:

- Planificación: La planificación del ciclo de trabajo se realizará el primer día de cada ciclo de trabajo y se verá reflejada como una lista de tareas a realizar en una columna especifica del tablero.
- Reuniones diarias: Las reuniones diarias no se llevaran a acabo. Para reemplazarlas se mantendrá una bitácora donde se llevará registro de cada día trabajado y lo realizado ese día.
- Retrospectiva: La retrospectiva se realizará el ultimo día del ciclo de trabajo y se representará como una lista de puntos a mejorar para el siguiente ciclo de trabajo.

Los objetivos de cada ciclo de trabajo serán definidos durante la planificación de cada ciclo de trabajo, no obstante, para conceptos de la planificación de fechas,

se han definido objetivos tentativos de cada ciclo de trabajo. La duración de los ciclos de trabajo será de 2 semanas cada uno.

4.1. Metodología de evaluación del proyecto

4.1.1. Experimentos controlados

Para la evaluación del proyecto en los ámbitos de facilidad de uso y simplicidad; se utilizarán experimentos controlados. Los experimentos controlados se caracterizan porque en ellos se controlan todas las variables que afectan al resultado, excepto una. Esta variable independiente permite que se evalúen los distintos efectos en muchos ambientes similares.[16]

Aplicado a nuestro caso, el software será presentado en las mismas condiciones a un grupo de usuarios. Los usuarios en este caso representarán las variables independientes en el experimento, ya que ellos harán que los resultados del experimento varíen unos de otros.Para evaluar se definirán una serie de objetivos que los usuarios deben cumplir y luego se utilizará un cuestionario para hacer el seguimiento.

El cuestionario para evaluar el cumplimiento de los objetivos será definido con posterioridad. Se planea utilizar la escala de Likert como métrica para dicho cuestionario.

4.1.2. Benchmarking

Esta evaluación apunta a verificar el cumplimiento de objetivos correspondientes a indicadores de computabilidad, como por ejemplo rapidez". A continuación se detalla cada indicador, su forma de comprobación y su resultado esperado:

(Las métricas de estabilidad, rapidez, seguridad, simplicidad y portabilidad están detalladas en la sección de Metodología de Evaluación)

- Estabilidad del servidor: La estabilidad del servidor será medida respecto a la cantidad de horas que puede mantenerse activo sin necesidad de ser reiniciado o incurrir en errores. Esto se realizará utilizando un cronometro, el cual será implementado directamente en el servidor. El valor esperado para este punto es de 24 horas de operación sin errores. *Se descartan errores o caídas producidas por consecuencia de errores de hardware o elementos externos.*

- Rapidez del servidor: La rapidez del servidor va en directa relación con la cantidad de conexiones que logra soportar y el tiempo de respuesta bajo una

carga moderada. Para esta evaluación se implementará una métrica del tiempo de conexión entre cliente y servidor, y se aplicará de manera artificial una carga a este mismo. Se espera que para una conexión entre Chile y EEUU, el tiempo de conexión este bajo los 300 milisegundos. *Se descartan peaks de latencia producidos por factores externos. La carga a generar en el servidor dependerá del hardware sobre el que se esté trabajando y que será definido con posterioridad.*

- Rapidez del cliente: La rapidez del cliente tiene que ver con la velocidad en la que este es capaz realizar la renderización del estado del juego. Para esto se medirá la cantidad de imágenes por segundo que es capaz de generar. Se espera que este valor esté por sobre los 60 fps. *Hardware de prueba: i3-7100, 8 Gb Ram, Nvidia GTX 1060 3 Gb*
- Seguridad: La seguridad del cliente se basa en una comunicación cifrada entre cliente y servidor. Y además en el uso de un servidor autoritario por sobre las acciones de los clientes. Esto se medirá con un Si o No, dependiendo de si se implementa o no seguridad en el canal de comunicación y de si se utiliza una arquitectura con servidor autoritario.[6]
- Portabilidad: La portabilidad tiene que ver con la capacidad de las distintas aplicaciones de ser utilizadas en distintas plataformas. Para este punto se evaluará con Si o No, en el caso que sea posible o no portar las distintas aplicaciones según las tecnologías que fueron seleccionadas en su implementación.

5. Plan de trabajo

A continuación se presenta el plan de trabajo. Este plan contempla 4 etapas. Para el caso de los ciclos de trabajo (sprints), el tema a revisar en cada sprint está sujeto a cambio, ya que será evaluado al comienzo de cada sprint conforme a las tareas pendientes y los nuevos desafíos que aparezcan durante el desarrollo. Las fechas están sujetas a reajustes.

Etapa 1: Concepción del proyecto

Actividad	Fecha de inicio	Dias
Investigación de tecnologías: Protocolos de comunicación	19 de Marzo	7
Investigación de tecnologías: Renderizado 2D en HTML5	26 de Marzo	7
Investigación de tecnologías: ORM para .Net Core	2 de Abril	7
Presentación de propuesta (Diapositivas)	9 de Abril	7

Etapa 2: Documentación del proyecto

Actividad	Fecha de inicio	Dias
Desarrollo de planificación	16 de Abril	7
Documento de formulación	23 de Abril	8
Documento de formulación (Entrega final)	30 de Julio	7
Documento de memoria	13 de Agosto	120
Primera revision documento de memoria	10 de Diciembre	7
Segunda revision documento de memoria	17 de Diciembre	7
Documentacion de uso	10 de Diciembre	14

Etapa 3: Implementación (Paralelo a la etapa 2)

Actividad	Fecha de inicio	Dias
Sprint 0: Historias de usuario, ambiente y arquitectura inicial.	16 de Abril	14
Sprint 1: Modulo de mapas y zonas	30 de Abril	14
Sprint 2: Editor del engine y gestor de sprites base	14 de Mayo	14
Sprint 3: Editor de mapas y zonas	28 de Mayo	14
Sprint 4: Modulo de personajes	9 de Julio	14
Sprint 5: Editor de clases	23 de Julio	14
Sprint 6: Modulo de chat y autenticacion	6 de Agosto	14
Sprint 7: Modulo de habilidades	20 de Agosto	14
Sprint 8: Modulo de interacción (Area of interest)	3 de Septiembre	14
Sprint 9: Modulo de interacción (Movimiento y colision)	17 de Septiembre	14
Sprint 10: Modulo de interacción (Efectos de habilidades)	1 de Octubre	14
Sprint 11: Editor opciones generales	15 de Octubre	14

Etapa 4: Evaluación

Actividad	Fecha de inicio	Dias
Pruebas de rendimiento	29 de Octubre	21
Experimento controlado	29 de Octubre	21

Referencias

[1] *Motor de videojuego*. URL: https://es.wikipedia.org/wiki/Motor_de_videojuego. (Visitada el 29 de Abril del 2018).

[2] *Gráficos 2D*. URL: https://en.wikipedia.org/wiki/2D_computer_graphics. (Visitada el 29 de Abril del 2018).

[3] *MMORPG*. URL: https://es.wikipedia.org/wiki/Videojuego_de_rol_multijugador_masivo_en_l%5C%C3%5C%ADnea. (Visitada el 29 de Abril del 2018).

[4] *Renderización*. URL: https://es.wikipedia.org/wiki/Renderizaci%5C%C3%5C%B3n. (Visitada el 29 de Abril del 2018).

[5] *Scrum*. URL: https://es.wikipedia.org/wiki/Scrum_(desarrollo_de_software). (Visitada el 29 de Abril del 2018).

[6] *Modelo cliente-servidor*. URL: https://en.wikipedia.org/wiki/Client%5C%E2%5C%80%5C%93server_model. (Visitada el 29 de Abril del 2018).

[7] *RPG Maker*. URL: https://es.wikipedia.org/wiki/RPG_Maker. (Visitada el 29 de Abril del 2018).

[8] *Harvest Online*. URL: https://www.kickstarter.com/projects/1193941746/harvest-online?ref=category&ref=discovery&term=mmorpg. (Visitada el 29 de Abril del 2018).

[9] *Eclipse Wikia*. URL: hhttp://eclipse-ge.wikia.com/wiki/Eclipse_Origins. (Visitada el 29 de Abril del 2018).

[10] *Eclipse Origins*. URL: https://www.eclipseorigins.com/. (Visitada el 29 de Abril del 2018).

[11] *Intersect Engine*. URL: https://www.ascensiongamedev.com/topic/691-intersect-development-road-map/?page=1. (Visitada el 29 de Abril del 2018).

[12] *Mystalia Engine*. URL: https://github.com/krazyjakee/MystaliaEngine. (Visitada el 29 de Abril del 2018).

[13] *Dot World Maker*. URL: https://www.dotworldmaker.com/index.html. (Visitada el 29 de Abril del 2018).

[14] *MMORPG Engine*. URL: https://github.com/LazyPlanet/MMORPG-Engine. (Visitada el 29 de Abril del 2018).

[15] *Trello*. URL: https://trello.com. (Visitada el 29 de Abril del 2018).

[16] *Experimentos Controlados*. URL: https://en.wikipedia.org/wiki/Scientific_control#Controlled_experiments. (Visitada el 29 de Abril del 2018).

[17] Marios Assiotis y Velin Tzanov. "A Distributed Architecture for MMORPG". En: ().

[18] Jianwei LiHualei WuXiaowen LiShixi Chen. "Network Synchronization Mechanism Design Based on MMORPG". En: ().

[19] Jens Muller y Sergei Gorlatch. "A Scalable Architecture for Multiplayer Computer Games". En: ().

[20] Jean-Sebastien Boulanger. "Area of Interest Management for MMO". En: ().