3

Análisis e Implementación de Arquitectura de un MMORPG para Celulares

Analysis and Implementation of MMORPG Architecture for Mobile Phones

Christian Devetak Cruz / Iván Francisco Silva Feraud

Resumen

El presente estudio realiza un análisis de diferentes arquitecturas asociadas con la creación de juegos de video. Como resultado del análisis, se definieron algunos factores para la creación de una arquitectura como: facilidad de implementación, escalabilidad, seguridad, carga para el servidor, tiempos de respuesta de la conexión y la fiabilidad de la interconexión. La topología definida para la arquitectura propuesta es híbrida, dado que utiliza los beneficios de cliente servidor y P2P. Como siguiente paso, se implementa la arquitectura propuesta mediante la creación de un juego de video de género MMROPG multiplataforma, para luego realizar mediciones de latencia, cuadros por segundo y estrés del servidor, y poner a prueba la usabilidad de la arquitectura. El juego resultante, basado en la arquitectura lograda tras el análisis comparativo, prueba ser funcional tanto en móviles como en computadores, y logra velocidades aceptables de conexiones, de tazas de actualización, y de estrés del servidor.

Palabras Clave:

MMORPG, arquitectura, multijugador, masivo, juego en línea.

Abstract

The aim of this study is to perform a comparative analysis of different architectures associated with the implementation of video games. As a result, some factors were established to propose the architecture, such as: easiness of implementation, scalability, security, server load, connection response time and interconnection reliability. The topology defined for the proposed architecture is hybrid, since it uses the benefits of client server and P2P. Consequently, a multiplatform MMROPG videogame was developed using the proposed architecture and a usability test was carried out by means of latency, frames per second and server stress time. At the end, the developed game resulted to be functional for both mobiles and computers, reaching promising results for critical measures like speed connection, refresh rates and server stress times.

Keywords:

MMORPG, architecture, multiplayer, massive, online, game, Java, framework.

Fecha de recepción: 04 de febrero del 2016
Fecha de aceptación: 06 de septiembre del 2016

ISSN: 1390 - 6399

Introducción

Un juego de video es una forma de arte interactiva en la que existe una respuesta visual y auditiva a acciones efectuadas por el usuario. La creación de un juego de video involucra varias habilidades como lo son conocimientos teóricos de la arquitectura de un juego, programación avanzada, multimedia, gráficos en dos o tres dimensiones, modelado, ingeniería de sonido, escritura de una historia, manejo de proyectos, lógica de diseño, física e inteligencia artificial por citar los más relevantes (Hussein, 2008).

Con el nacimiento de las redes de computadora también lo hacen los juegos multijugador, en los cuales dos o más usuarios en terminales diferentes conectadas entre sí. Con el mejoramiento del internet, se abrió la posibilidad de juegos Multijugador Masivos en línea o MMOs por sus siglas en inglés.

Un MMO típico posee un mundo virtual persistente donde miles de jugadores se pueden integrar a una partida o abandonarla, sin que el mundo o el resto de jugadores se ven afectados (Bjørlo & Voll, 2009). Los MMO se los puede clasificar por géneros como:

- MMO Role Playing game – Juego de Rol Multijugador Masivo Online (MMORPG).
- MMO First Person Shooter – Juego de disparos Multijugador Masivo Online (MMOFPS).
- MMO Real Time Strategy – Estrategia en tiempo real Multijugador Masivo Online (MMORS).
- MMO Sport Game – Juego de Deporte Multijugador Masivo Online (MMOSG).

De todos ellos, el género más exitoso es el de MMORPG, donde un gran número de jugadores forma comunidades virtuales (Chan & Chang, 2004; Suznjevic, Dobrijevic, & Matijasevic, 2009) - no solo interactúan con el entorno, si no además con los personajes de otros jugadores, y con el mundo virtual, teniendo objetivos definidos (Suh, Kim, & Kim, 2010). En este género, los jugadores exploran y aprenden sobre el mundo y los objetos que los rodean.

De acuerdo a datos y estimaciones de SuperData en el 2013, se espera que para el 2017 existan en el mundo unos 50 millones de jugadores de este género, invirtiendo 12.8 billones de dólares al año (SuperData Research Inc, 2015). Esto implica que hay una gran demanda a nivel mundial de este género de juegos de video. Sin embargo, diseñar un MMORPG y ponerlo en marcha no es un trabajo sencillo pues involucra múltiples partes que deben encajar entre sí.

Por la cantidad de factores que deben ser considerados para construir un juego de video, cada uno de ellos realiza estudios apoyados con diagramas llamados "diseños de estados" que explican cómo deben ir conectadas las diferentes piezas. A su vez, los diseños de estado de cada una de las partes del software se acoplan a diseños más grandes que incluyen interconexiones entre software y/o hardware y a servicios adicionales, a lo que llamamos una arquitectura.

En este estudio, se realizará un análisis comparativo de diferentes arquitecturas para llegar un esquema ideal que permita la construcción de un juego de video MMORPG multiplataforma, teniendo en cuenta la facilidad de implementación, escalabilidad, seguridad, carga para el servidor, tiempos de respuesta de la conexión, y la fiabilidad de la interconexión. Finalmente, se implementará la arquitectura propuesta mediante la creación de un juego de género MMROPG multiplataforma, en el cual se realizarán mediciones de latencia, cuadros por segundo y estrés del servidor, para poner a prueba la usabilidad de la arquitectura.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

Este artículo está dividido en cinco secciones. La primera sección brinda conceptos sobre varios componentes de un MMORPG e introduce las arquitecturas a ser analizadas. En la segunda sección se muestran análisis comparativos de las arquitecturas y expone una arquitectura propia. La tercera sección detalla la metodología utilizada para realizar pruebas sobre el desempeño de la arquitectura y del juego resultante. En la cuarta sección, se muestran los resultados de las evaluaciones. Finalmente, en la quinta sección, se relatan las conclusiones del estudio.

Marco Teórico

Diseño de un MMORPG.

Diseñar un MMORPG requiere la conexión de partes para lograr que todo funcione. Las partes más importantes del diseño MMORPG, se componen módulos multimedia los cuales contemplan motores gráficos (Yan-hui, Xia-xia, & Jin, 2011), arte del juego, mapeado y sonido, que son propios del cliente y cuyo rendimiento depende de la terminal a ser utilizada; y de módulos de la arquitectura conformados por la programación, las interacciones entre usuarios y servidor, la inteligencia artificial, las bases de datos, las conexiones de red y la seguridad (Lee, Park, Kim, Youk, & Ryu, 2008).

Todo juego, sin importar su género característica cuenta con un ciclo llamado lazo principal o 'Main Loop'. Este contempla principalmente la lógica del juego y el pintado de la pantalla (Obviam, 2010; Yan-hui, Xia-xia, & Jin, 2011). Dentro de los lazos, se van invocando todos los demás módulos.

Wu, Huang & Zhang (2006) definen a la interacción entre usuario a usuario y de usuario a servidor como 'ciclo de transmisión'; el servidor envía información sobre el estado del juego (incluyendo estados de otros clientes) a cada cliente. Los clientes sincronizan los estados del servidor con sus estados locales individuales, procesan los comandos del jugador, y envían de vuelta las acciones del personaje (Suselbeck, Schiele, & Becker, 2009).

Los MMOPGS, además de los jugadores humanos, contienen NPCs -siglas para "Personaje no jugado" en inglés- que pueden ser tanto amigos como enemigos, utilitarios o simples objetos decorativos (Lee, Park, Kim, Youk, & Ryu, 2008).

Toda la información no volátil relacionada con el juego se guarda en una base de datos, que incluye cuentas de los usuarios, estados del juego, inventarios de objetos, estadísticas de jugadores, mapas, y demás datos dependiendo del juego (Chan & Chang, 2004).

Las conexiones de red son la forma en la que las terminales interactúan con el servidor y viceversa. Se utilizan por lo general 2 tipos de protocolos, el TCP y el UDP (Wu, Huang, & Zhang, 2006). Estas deben ser capaces de soportar muchísimas terminales conectándose y desconectándose a distintos tiempos. Debe de poderse escalar en ancho de banda, ser accesible y consistente (Chan & Chang, 2004). Adicionalmente, se debe considerar el ancho de banda, por lo tanto es vital tener en cuenta el tamaño de la carga y de los paquetes de datos a ser enviados (Cidon, Rom, Gupta, & Schuba, 1999).

Las conexiones deben ser seguras para garantizar la equidad en el juego, ya que existen herramientas para hacer uso indebido de la información enviada, diseños no seguros o errores de programación (bugs) para obtener algún beneficio, algo a lo que llamamos "exploits" (McGraw & Hoglung, 2007).

Los diseños del cliente, servidor, y demás partes anexas involucradas forman una Arquitectura. Algunas arquitecturas propuestas consideran el uso de varios

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

servidores "espejos" o clones del principal, los cuales balancean la carga. Otros proponen que las conexiones sean directas de cliente a cliente, siendo supervisadas por el servidor, algo a lo que se llama un híbrido. El problema con estos diseños es, que al no ser centralizados, resulta muy difícil una correcta sincronización de los estados de los jugadores (Chan & Chang, 2004).

Topologías.

Existen dos tipos de topologías básicas en las que una arquitectura de un juego multijugador se puede basar, la de Cliente-Servidor y la de P2P. El término P2P hace referencia a una conexión de "Persona a Persona".

En la topología P2P pura, no existe un servidor central administrando las conexiones. Cada terminal de cada usuario es responsable por cargar una parte del mundo virtual y de manejar las conexiones. Este esquema presenta tres problemas graves que lo vuelven inviable como base para un MMORPG (Bjørlo & Voll, 2009):

- Si dos o más clientes se desconectan, el mundo virtual se ve dividido o se ve enteramente disuelto.
- No existe un punto principal de conexión, por lo tanto no hay una forma estable de conectarse al juego.
- No existe una autoridad central, posibilitando al cliente de hacer trampa.

Sin embargo, P2P puede prestar parte de sus beneficios, como lo son el uso de los clientes en su doble papel como servidores haciéndolo más ligero, o la reducción de latencia entre usuarios cercanos, al integrarlo con la topología de Cliente-Servidor, convirtiéndola en una topología híbrida.

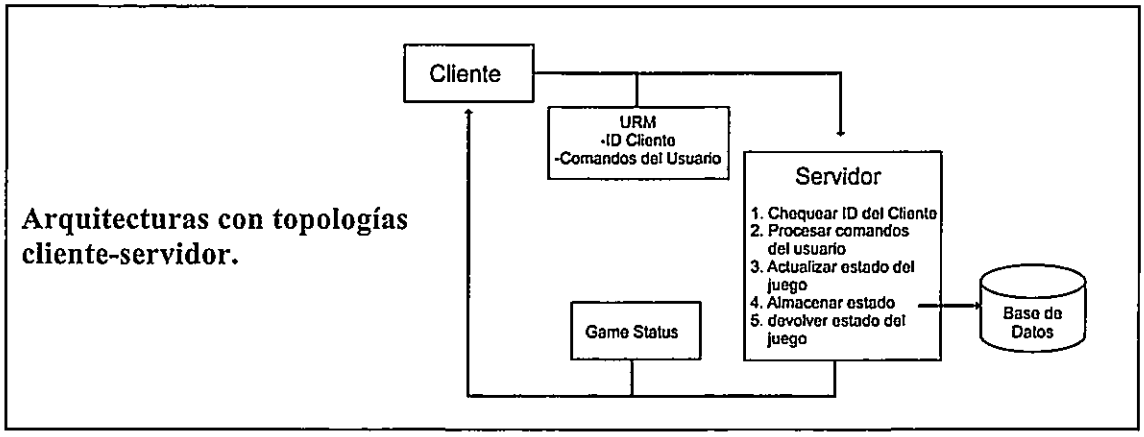

Figura 1. Diagrama operacional del servidor SSF basado en la arquitectura de Chan, H., & Chang, R. (2004).

En el artículo de Chan y Chang (2004) se analiza la arquitectura del juego Strife-shadow Fantasy (SSF). SSF es un juego multijugador masivo online que utiliza un modelo de cliente - servidor.

En la Figura 1 hay un servidor central que contiene el motor del servidor y la base de datos, se pueden conectar miles de clientes al servidor mediante su propia terminal. Chan y Chang dicen que este es quizás el mejor modelo para comenzar el desarrollo de un MMORPG, pudiendo ser escalable en el futuro, pues más servidores pueden ser agregados para balancear la carga.

SFF, al igual que todos los MMORPG, utiliza una serie de estados que indican al cliente como al servidor, el flujo del juego y que acciones tomar (ver Figura 2). El ciclo de los estados del juego para SSF son

ISSN: 1390 - 6399

validación del registro, ingreso del usuario, presentación del mundo virtual, un lazo de actualización del estado que verifica si el usuario ingreso comandos y los procesa, verificadores de finalización del juego y la salida del mismo.

Un factor clave de este modelo es la seguridad. Al estar la información en el servidor, y siendo este el único que la procesa y modifica (siendo el cliente un simple lector de dicha información), se tiene un mejor control y protección sobre la misma. Si un cliente necesita actualizar la información, esta petición pasa por el servidor el cual puede validar dicho requerimiento.

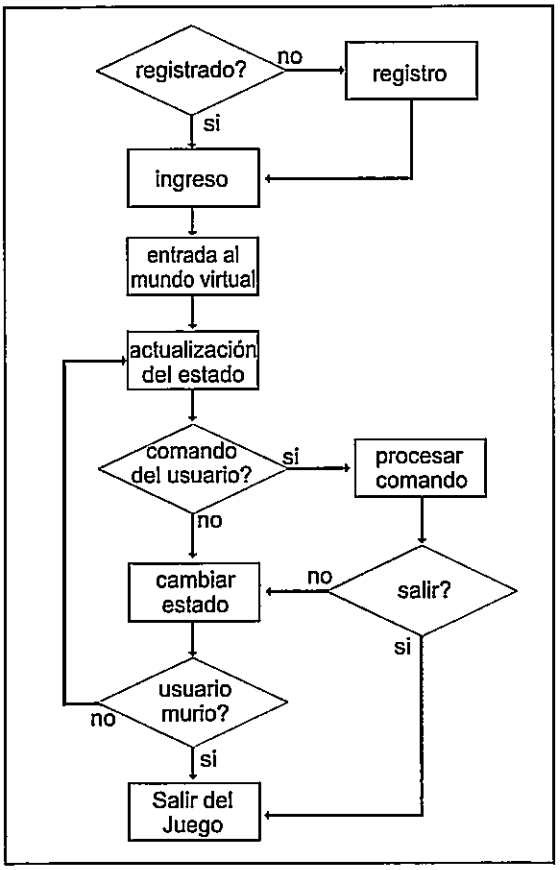

Figura 2. Estados del juego del cliente SSF, basado en la arquitectura de Chan, H., & Chang, R. (2004).

Por otro lado, el trabajo presentado por Arnold Hendrick, un veterano en la industria del juego con más de 25 años en el entorno, parte del desarrollo en más de 20 títulos y participe del famoso juego Marvel Superhero Squad Online, documenta bien en su website mmotidbits.com la arquitectura utilizada en sus proyectos.

Hendrick también habla de un modelo cliente-servidor en el que un software cliente, a través del internet se comunica a un servidor conformado por la inteligencia artificial y la base de datos como se puede observar en la Figura 3.

Este modelo resulta muy simple, todos los usuarios se conectan al servidor, y este maneja cosas como los personajes adicionales no controlados por los usuarios (NPCs), operaciones con la base de datos, y da proceso a los comandos, es decir, un esquema muy parecido al de SSF.

Hendrick además agrega módulos adicionales de soporte, manejo del servidor y análisis de métricas, que facilitan la ayuda, el mantenimiento y la optimización del mundo virtual.

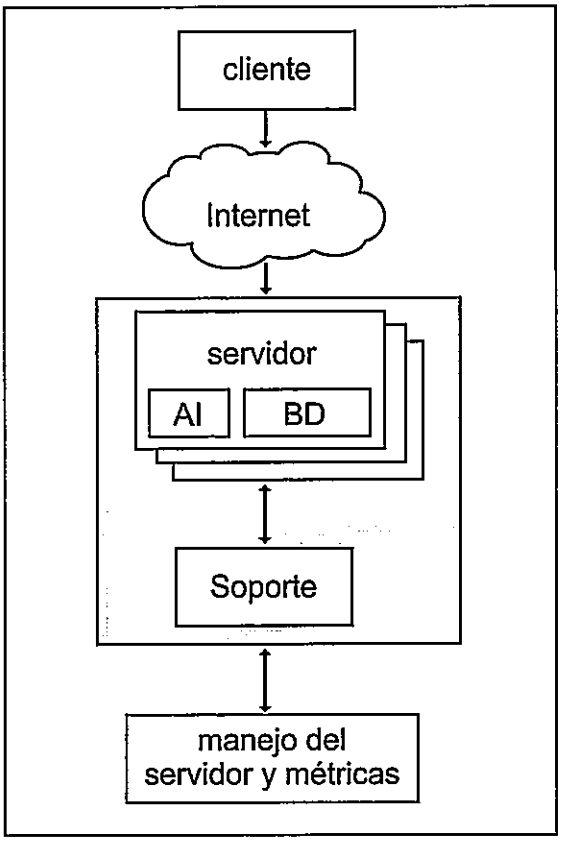

Figura 3. Arquitectura del modelo de hendrick. Basada en el modelo de Hendrick (2011).

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

Otro interesante caso de estudio es el de la arquitectura descrita por Hyun Sung Chu, asesor de ingeniero de software de IBM y arquitecto de varios proyectos Multijugador Masivos Online (MMO) para su empresa (2008).

Sung relata que la arquitectura de un MMO consta de 4 componentes principales: Un software cliente que presenta los gráficos al usuario, un servidor que interactúe con el cliente, servidor web de aplicaciones que se integre al cliente y al servidor, y una base de datos para almacenar información persistente.

Sung basa este modelo en el proyecto TGEA, siglas en ingles de Motor de Juego Avanzado Torque, el cual fue desarrollador para el juego "Torque 3D", pero también utilizado en otros juegos.

TGEA distribuye la carga en 3 servidores: El servidor TGEA (o servidor del juego) contiene el motor del juego e inteligencia artificial. El segundo servidor contiene un motor web para las páginas web del juego, y tercero, un servidor de bases de datos, el cual almacena toda la información del juego como se observa en la Figura 4.

Sung va un paso más allá y expone una arquitectura cliente-servidor de tipo zona. Cuando una arquitectura está estructurado de tal modo que distintas zonas del mundo virtual se encuentran en distintos servidores, a esto lo llamamos una solución "zonal". Por el contrario, en una solución "sin zonas", todas las áreas del mundo virtual están presentes en el mismo servidor (Wu, Gong, & Zheng, 2006; Bjørlo & Voll, 2009).

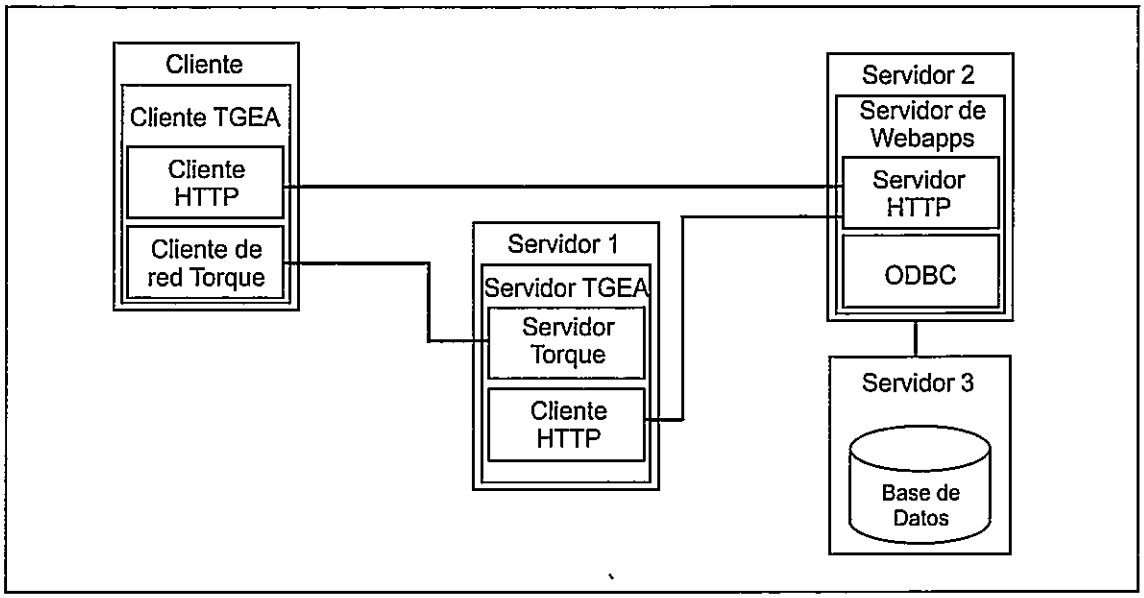

Figura 4. Arquitectura de TGEA (Sung, 2008).

El uso de zonas balancea aún mejor la carga en múltiples servidores, pero vuelve a la interacción entre sujetos de distintos servidores muy limitada o nula, además de compleja.

En el esquema de la Figura 5, se divide la carga en 2 servidores espejo. Estos cuentan con servidor web público con información del juego, y un servidor web seguro donde se ejecutan acciones que influencian sobre el servidor del juego, ambos conectados a la base de datos.

ISSN: 1390 - 6399

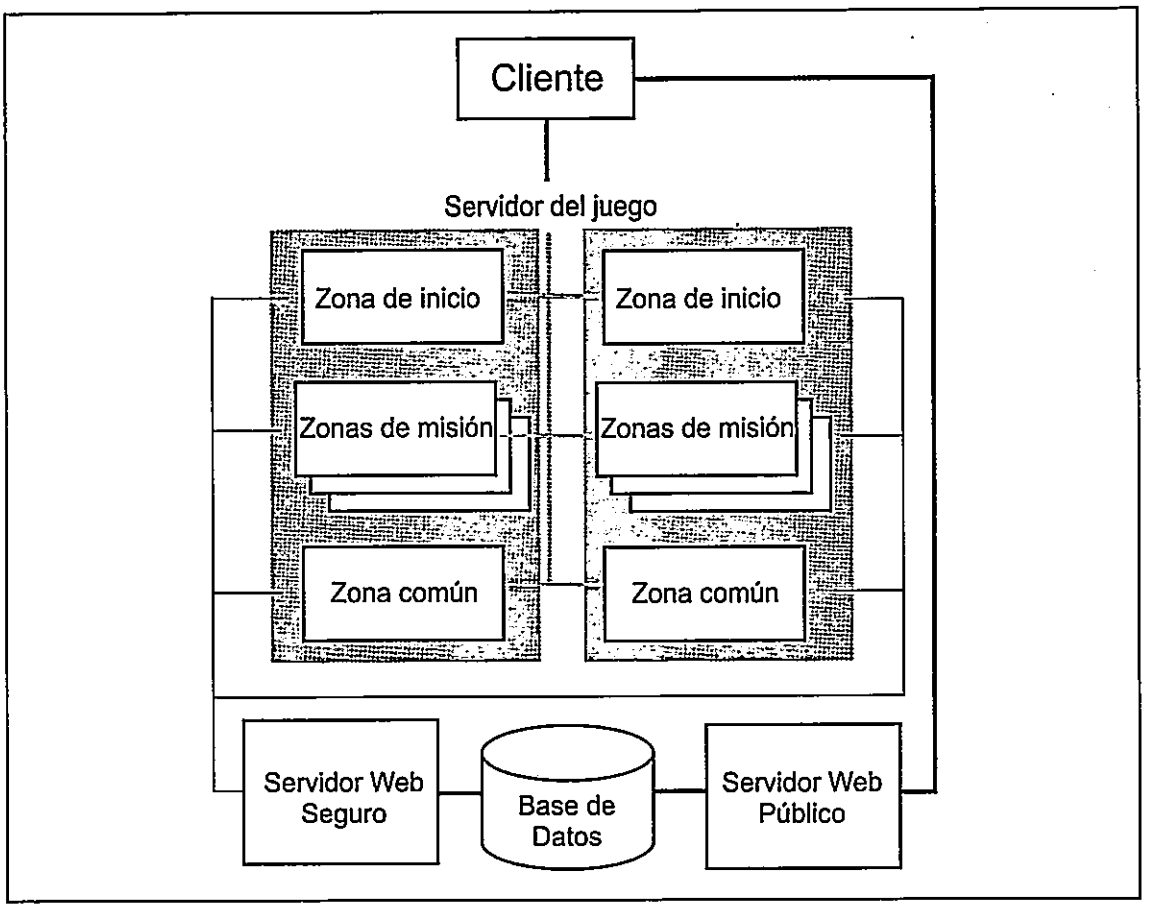

Figura 5. Arquitectura de TGEA en vista multizona (Sung, 2008)

Arquitecturas con topología híbrida.

Algunas arquitecturas relativamente nuevas, para reducir los recursos necesitados y el ancho de banda, implementan una arquitectura híbrida entre topologías P2P y cliente-servidor (Huey-Ing Liu, 2008).

Frode Voll Aasen y Tom-Christian Bjorlo Johannessen de la universidad de ciencia y tecnología de Noruega, basan su tesis en una arquitectura P2P y Cliente-Servidor hibrida.

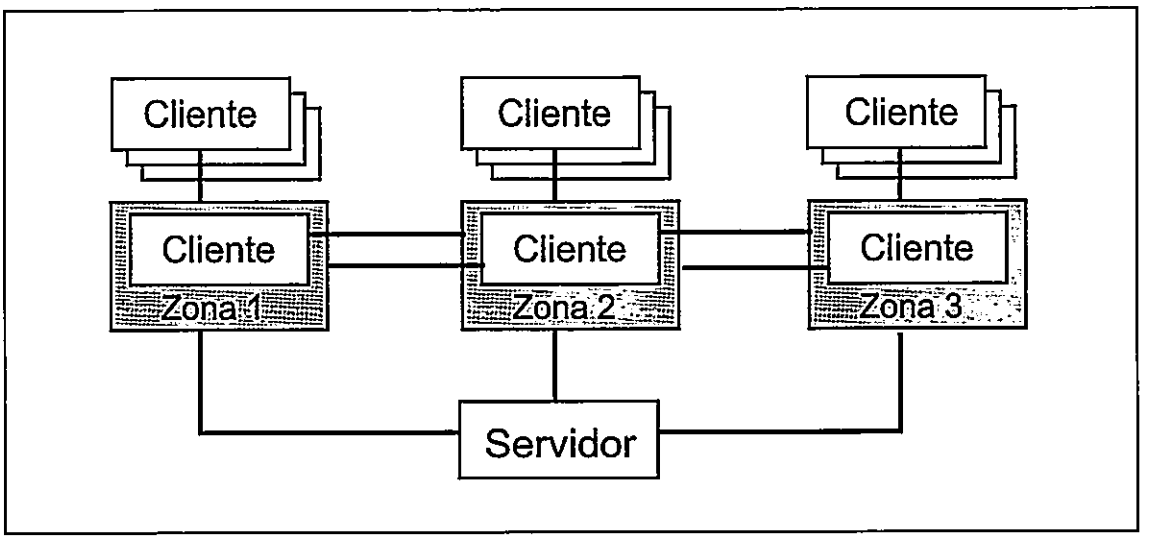

Figura 6. La arquitectura hibrida de 'tisu (Bjørlo & Voll, 2009).

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

La Figura 6 explica que esta arquitectura, la cual está siendo implementada en un juego de nombre "tisu", se realizó al asignar un cliente como administrador de una zona, es decir, el software de algunos clientes también actúa como un complemento del software servidor, el cual está constantemente sincronizando datos con el servidor real del juego (Bjørlo & Voll, 2009).

El mundo virtual de "tisu" está separado en varias zonas. El servidor asigna usuarios (clientes) para ser los administradores de cada zona, pudiendo asignar un nuevo administrador a una zona en cualquier momento. Los administradores se convierten en "supernodos" que reciben conexiones de los otros usuarios que ingresan en su zona.

Voll Aasen & Bjorlo exponen que el objetivo principal de esta arquitectura es el reducir costos de ancho de banda y servidores, al contar con parte del trabajo de procesamiento realizado por los clientes.

También explican que el sistema se vuelve complejo al necesitar interconexiones entre supernodos para permitir las transiciones interzonales, y reconocen que existen debilidades en este sistema en cuanto a la seguridad de la información en los supernodos.

Otra arquitectura con topología híbrida es la que expone el proyecto DaCAP, siglas en inglés para Arquitectura anti-trampas para P2P Distribuida.

DaCAP, proyecto propuesto por Huey-Ing Liu y Yun-Ting Lo de la Universidad Católica de Fu Jen, Taiwan, va un paso más allá e integra mejores seguridades para lidiar con el problema principal de esta topología, que es el estar propenso a trampas de los usuarios.

A medida que los MMORPGs aumentan en popularidad, también aumentan los estudios sobre los mismos. Las propiedades y objetos virtuales son objeto de estudio sobre el valor que los usuarios ponen sobre estos bienes (Shen, 2010; Kennedy, 2008).

"Los juegos online son un gran negocio. (...)En cualquier sitio donde hay dinero en juego, los criminales hacen de las suyas. En el caso de los MMORPG, los tramposos tienen un incentivo económico real para romper el juego" (McGraw & Hoglung, 2007).

Es por ello que el énfasis en la seguridad es sumamente importante. Al igual que el modelo de 'tisen, DaCAP también utiliza clientes como supernodos, pero áreas sensibles con gran número de usuarios o "puntos calientes" (como lo son por ejemplo los pueblos), son administradas directamente por el servidor. Dentro del servidor funcionan además módulos como servidor de sesiones, servidores de puntos calientes y de misiones, de área, de validación y la base de datos (Huey-Ing Liu, 2008).

DaCAP propone una solución para lidiar con los tramposos: cuando el cliente se encuentra conectado a un supernodo y no al servidor, todos los demás jugadores en el área se monitorean unos a otros. Si algún jugador hace trampa, el resto de clientes lo reportan al servidor de validaciones.

Metodología

En esta sección se analizarán los factores en común de las arquitecturas revisadas en el marco teórico, analizado, adicionalmente, la velocidad de los protocolos. Basado en dicho estudio, se propondrá una arquitectura, según los beneficios o ventajas resultantes del análisis. Finalmente, se desarrollara un prototipo de juego MMORPG al que llamaremos en su versión alfa "Alos Online", para poder realizar las pruebas necesarias y medir la arquitectura.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

Análisis de las arquitecturas.

Se analizaron un total de cinco arquitecturas, de las cuales se discernió seis factores principales y en común en todas ellas: simplicidad, escalabilidad, carga del servidor (ligereza), interconexión, latencia y seguridad.

La arquitectura de SSF es muy completa y documentada, pero se vale del Protocolo de Transferencia de Hyper-Texto (HTTP) para su comunicación entre cliente servidor. Este protocolo fue desarrollado para la transferencia de datos de las páginas de internet. Una desventaja del mismo es que el servidor no puede "empujar" mensajes hacia el cliente, obligando al cliente a preguntar si existen cambios en intervalos de tiempo. Se realizó una prueba entre los protocolos TCP, UDP, y HTTP para comprobar sus velocidades como se puede observar en la Figura 7.

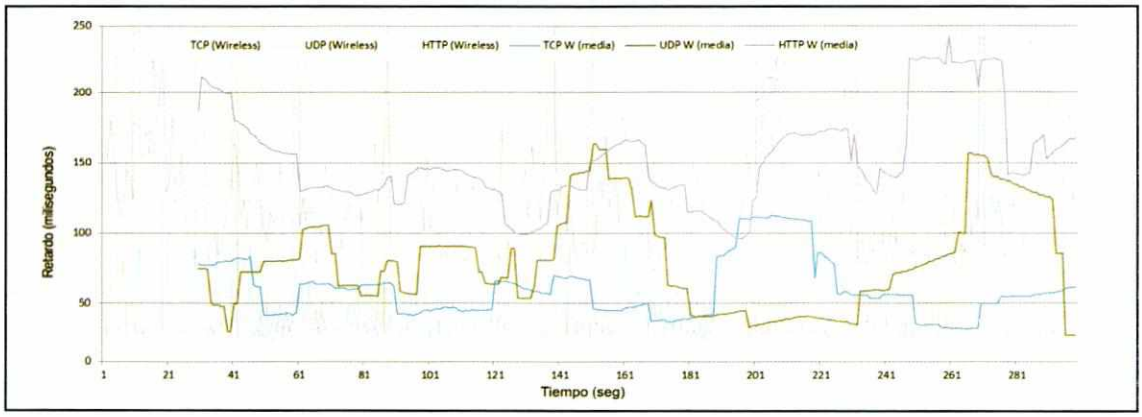

Figura 7. Comparación de velocidades entre HTTP, TCP y UDP sobre una red Wireless.

La prueba mostro que el protocolo TCP mostro ser la más rápido sobre WIFI con un promedio de 59 milisegundos, en segundo lugar UDP con 79 ms, y en tercer lugar HTTP con 155 ms. Sin embargo, una razón lógica por la cual UDP no supero a la velocidad de TCP fue que para efectos de la prueba, se necesitó programar un acuse de recibo para el protocolo UDP, el cual carece del mismo, para que así el cliente pueda saber cuándo enviar el siguiente paquete.

Se utilizó una red móvil 3G para la segunda prueba, obteniendo como resultado tiempos de respuesta promedio de 78 milisegundos con el protocolo TCP, 139 milisegundos con UDP y 267 milisegundos con HTTP.

Por el resultado, ver Figura 8, TCP será el protocolo a ser usado para la comunicación de la arquitectura planeada, integrando también UDP para información no crítica y continua como el desplazamiento de los usuarios sobre el mapa.

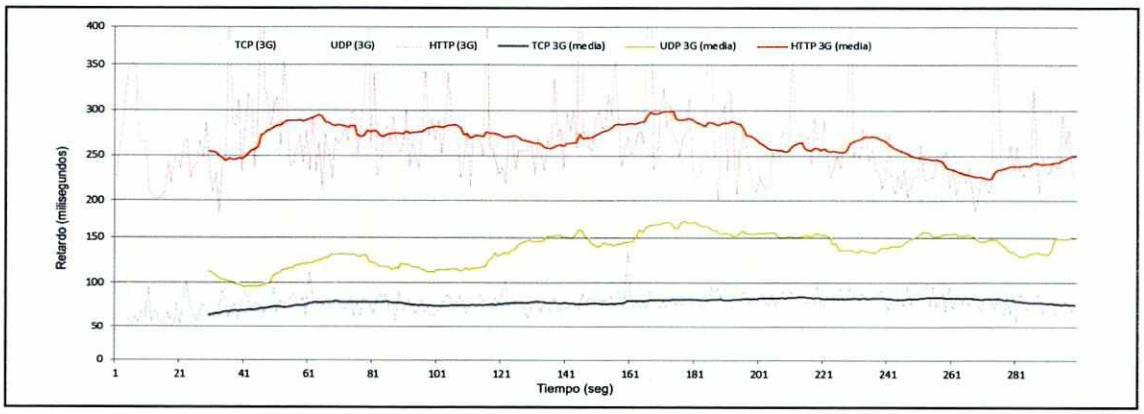

Figura 8. Comparación de velocidades entre HTTP, TCP y UDP sobre una red 3G.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

Para la selección de la mejor arquitectura se realizó una comparación de las fortalezas de cada una de ellas como se observa en la Figura 9. El modelo de Hendrix es probablemente el más fácil de implementar de todos. Es sin embargo muy básico y el menos documentado. Se trata de una base que pudo ser referente para Strifeshadow o para TGEA es muy completo, la única falencia es la complejidad de implementar zonas, la cual puede ser obviada en las primeras etapas de un MMORPG.

Los últimos dos modelos analizados, de topología híbrida pueden quedar descartados. Es indiscutible, en términos de seguridad y de nivel de dificultad de implementación, que la mejor topología es la de cliente-servidor.

Es por eso que la mayoría de MMORPGS utilizan la arquitectura Cliente-Servidor, aunque presenta claras desventajas en cuanto a poder de procesamiento y de recursos utilizados por el servidor, es más segura, de fácil implementación y escalabilidad, tres factores claves a la hora de implementar y mantener un MMORPG.

El rendimiento del servidor es un problema latente, la tecnología avanza a pasos gigantescos y permite la adopción de mejores plataformas conforme el juego va evolucionando y el número de usuarios se va incrementando.

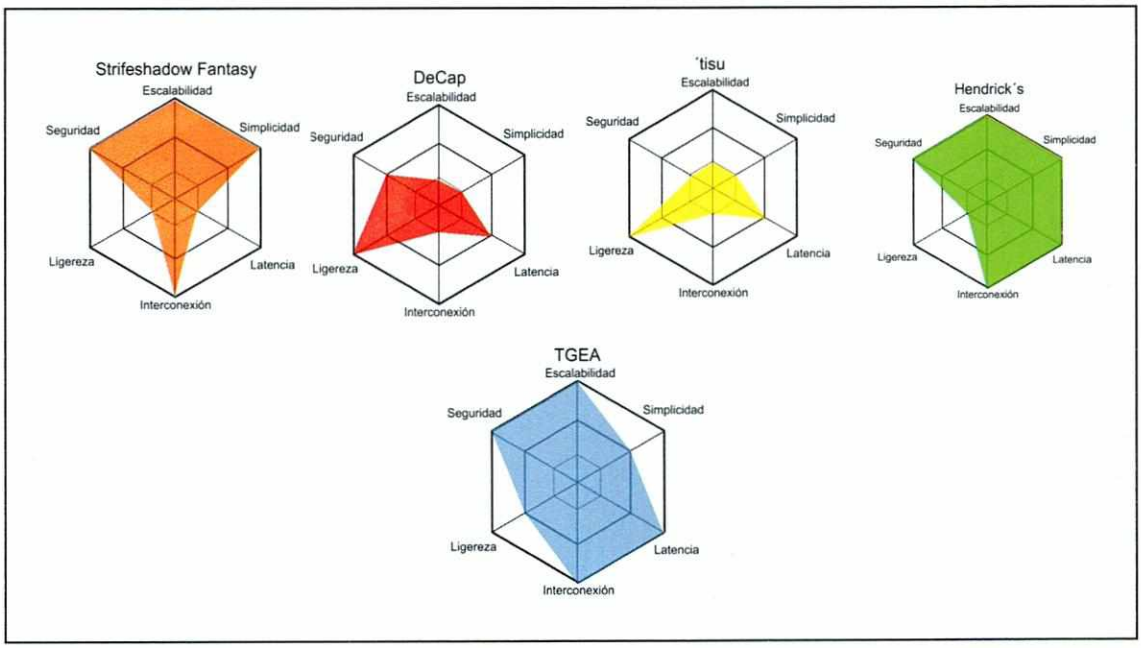

Figura 9. Hexágonos comparativos de las fortalezas de cada arquitectura.

Aunque DeCAP hace un esfuerzo en el tema de seguridad, no puede garantizar la validez de los datos si todos los usuarios en un área han manipulado los datos para su beneficio. Aun así, ciertos aspectos son rescatables del modelo, como el reporte automático de otros usuarios en el área sobre actividad ilegal de algún usuario haciendo trampa, o como el destinar ciertos servicios a otros servidores.

Arquitectura propuesta.

En la Figura 10 se puede observar la arquitectura propuesta del juego que llevará el nombre de "Alos Online" en su versión de desarrollo, y contará con una topología cliente-servidor. La arquitectura estará basada en el modelo de Hendrick's e implementa varios de los módulos de Strifeshadow Fantasy, pero con protocolos TCP y UDP.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

El servidor cuenta con dos servicios: el Servidor de Alos y el servidor web Apache. Ambos servicios disponen de su propia base de datos.

El servidor de Alos utiliza módulos para el ingreso de los usuarios, conexiones, validaciones en temas de seguridad, sirve los diversos mapas que conforman el mundo virtual, la inteligencia artificial de los NPCs y del entorno, tiene estados de memoria con los ítems que se encuentran en juego.

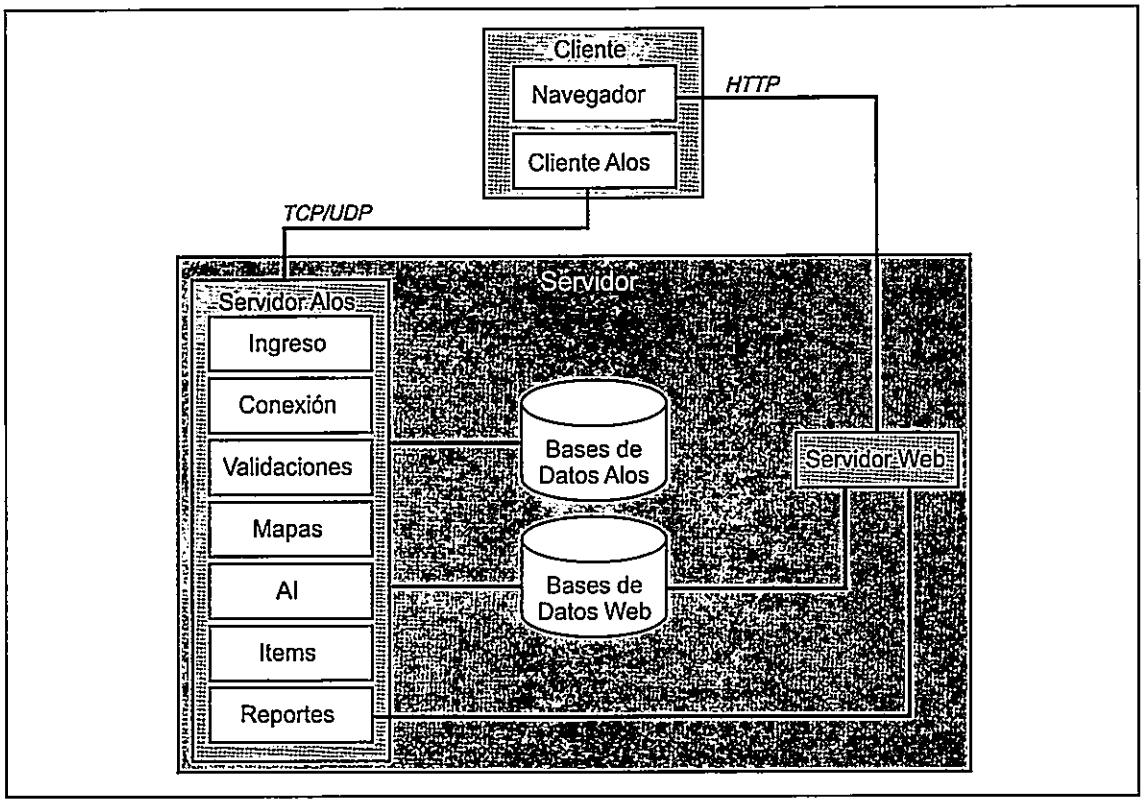

Figura 10. Arquitectura propuesta para Alos Online

El cliente posee todos los gráficos que el mundo virtual necesitará para poder formar los mapas y poner en ellos los personajes, NPCs y demás objetos.

Implementación de la arquitectura en el desarrollo de Alos.

Tanto el software cliente como el software operando en el servidor están compuestos de una serie de módulos, cada uno con una tarea específica. El conjunto de las tareas forman un diseño que hace que la arquitectura funcione.

El cliente de Alos implementa al framework libGDX la librería Kryonet, la cual permite él envió y la recepción de paquetes TCP y UDP. El primero transmite toda la información relevante del juego como ítems, estados etc., y el segundo el desplazamiento de los personajes sobre el mapa.

Las funciones principales del cliente son:

- Conectar al cliente con el servidor
- Proveer de una interface gráfica para el usuario UI o GUI.
- Recibir los comandos del usuario
- Enviar los comandos del usuario al servidor.
- Recibir instrucciones del servidor y convertirlas en respuestas visuales.

Tras la autenticación de ingreso al juego utilizando el cliente "Alos", se recibe toda la información que el cliente necesita para ubicarnos por primera vez en la pantalla.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

En la Figura 11, observamos que el lazo principal también recibe actualizaciones del servidor y escucha los eventos del usuario (clicks, toques de pantalla, y demás métodos de entrada).

El lazo principal corre a una velocidad mesurable ya sea por cuadros por segundo FPS o por Actualizaciones por segundo UPS.

Los UPS son las acciones realizadas por el actualizador de estados, que es el que contiene toda la lógica del cliente.

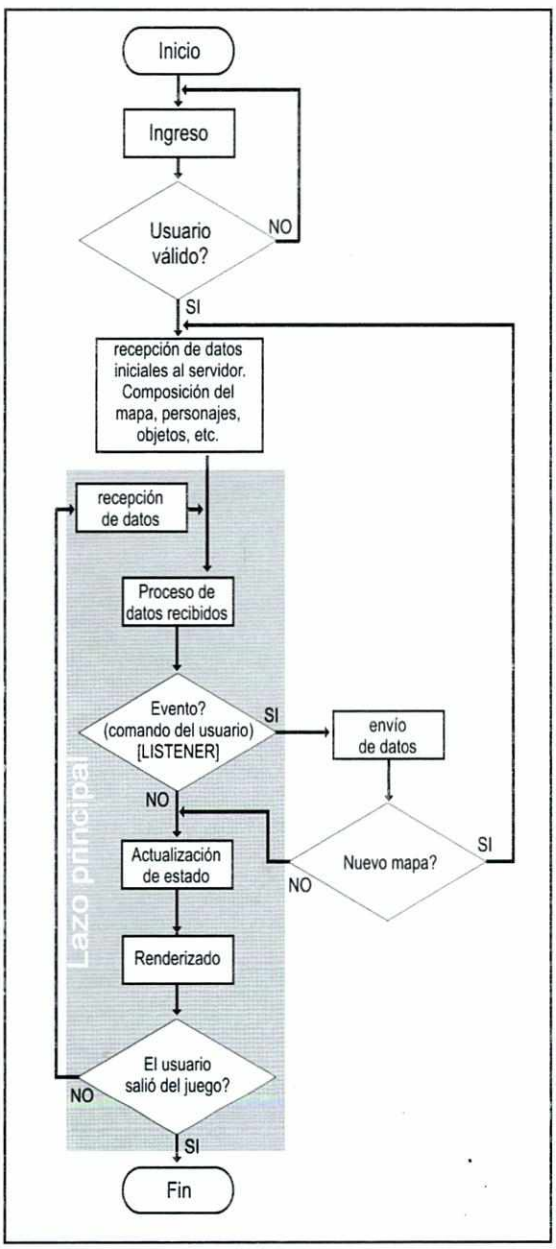

Figura 11. Diseño de estados del programa cliente de Alos Online

Los FPS indican el número de veces que la pantalla es actualizada o pintada, lo que en juegos de video normalmente ocurre en ciclos de 3 a 60 FPS.

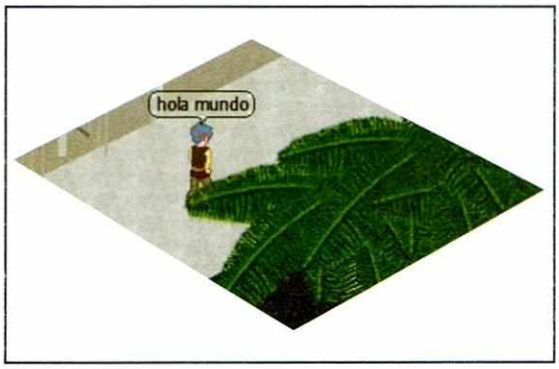

Figura 12. Gráficos isométricos

Alos utiliza graficas en 2D en un diseño isométrico como se muestra en la Figura 12. La isometría es una técnica utilizada por algunos juegos (Age of Empires, SimCity, etc) que permite una mejor reutilización de gráficos, pues solo se necesitan perspectivas de frente y de espaldas para cada elemento, invirtiendo la gráfica horizontalmente según se requiera.

El mapa está formado por pequeñas baldosas. El servidor da instrucciones al cliente sobre la composición del mapa para que el cliente lo renderize utilizando las baldosas adecuadas. Los mapas son creados con una utilidad de libre descarga en el internet llamada Tiled Map Editor como se muestra en la Figura 13.

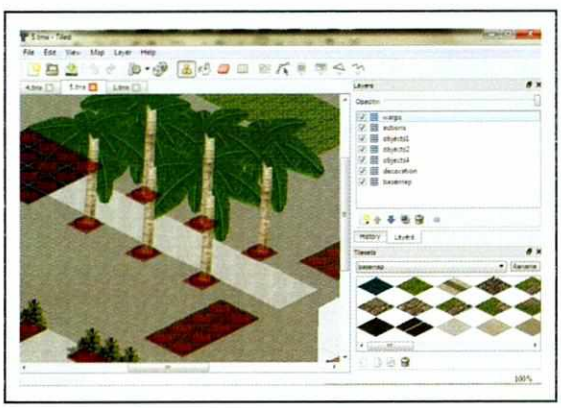

Figura 13. Editor de mapas Tiled

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

Otro componente clave es el "escuchador" o listener, el cual está pendiente de las acciones que el usuario realiza, y ordena al actualizador del juego el realizar los cálculos necesarios para que el renderizador pueda mostrar al usuario el resultado.

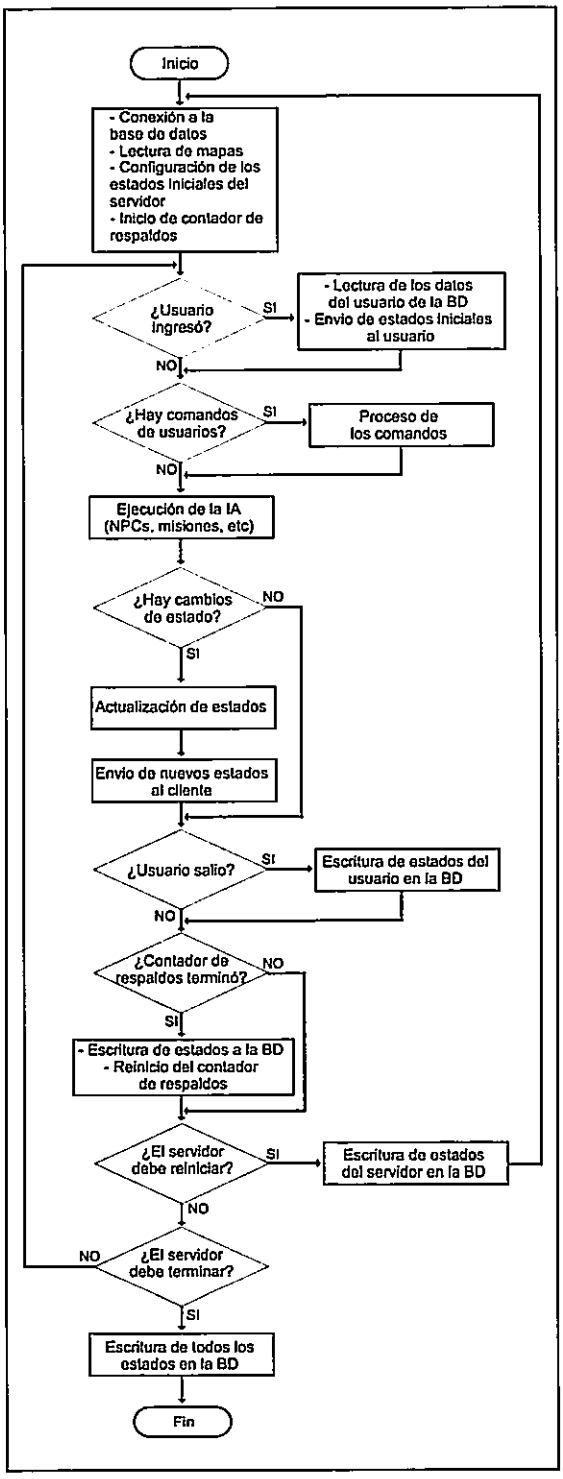

Figura 14. Diseño de estados del servidor de Alos Online

El servidor de Alos, es el encargado de almacenar la información de todos los jugadores y del juego mediante el uso de una librería de conexión a los clientes y una a la base de datos MySQL. Al iniciar el servidor, este recoge toda la información que necesita para operar de la base de datos y de archivos de texto.

Cuando un mapa es creado o modificado con el programa Tiled, el servidor lee la fecha de modificación del mapa, un interpretador de XML se encarga de extraer la información y almacenarla en base de datos y en archivos de texto.

Dentro del ciclo del servidor, este también se pone a la escucha de cualquier comando proveniente de cualquier jugador registrado. Si existen cambios de estado (lo cual es lo más común que suceda), el servidor envía la información a todos los usuarios involucrados directa o indirectamente con el cambio como se observa en la Figura 14.

Para la comunicación entre el cliente y servidor se utilizan paquetes de texto plano encriptados separados por comas y otros caracteres, lo cual reduce el tamaño de los paquetes. Los paquetes son enviados asincrónicamente mediante el uso de hilos.

El esquema en la Figura 15 muestra una negociación típica entre un cliente y el servidor. El cliente envía comandos al servidor mediante el uso de paquetes, a los que el servidor procesa y entrega una respuesta. El servidor también envía información de actualización al cliente sin que este la solicite.

La arquitectura de Alos del lado del cliente implementa un framework de nombre LibGDX al que se agregan clases y librerías que permiten la existencia de módulos para realizar las distintas tareas requeridas:

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

- Clase principal, la cual inicia el juego, configura variables de inicio y corre el laso principal.
- "Listener" de entrada de información del usuario.
- Clase que maneja la conexión y recepción de mensajes.

Además, el cliente corre en otro hilo, la librería Kryonet que contiene las clases necesarias para las conexiones y manejo de paquetes.

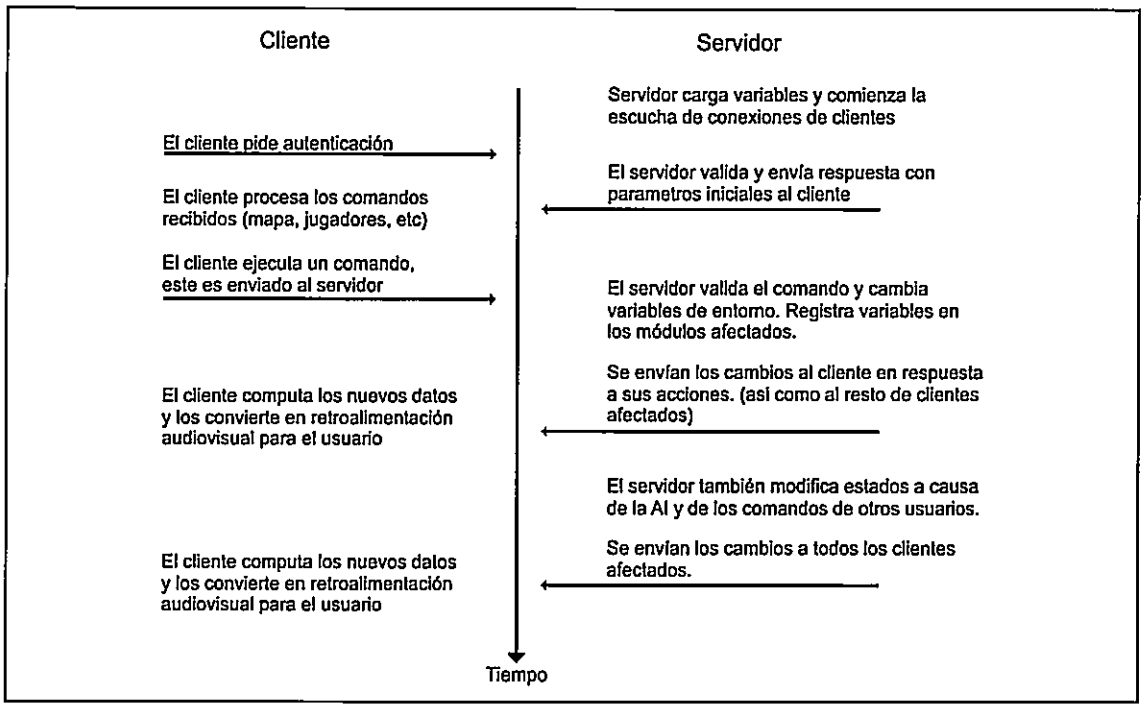

Figura 15. Modelo típico de la conexión entre el cliente y el servidor de Alos Online

Del otro lado de la conexión tenemos al servidor Alos, el cual también cuenta con dos hilos principales:

El primer hilo corre la librería Kryonet. Esta, al igual que en el cliente, se encarga de gestionar la conexión, pero también involucra lógica del juego mediante el uso de los siguientes componentes:

- La clase servidor, se encarga de inicializar los estados del juego.
- La clase paquete, es la encargada de sostener las variables que serán enviadas a los clientes.
- El "network listener", gestiona las conexiones y escucha los paquetes recibidos. Da trámite a los requerimientos de los usuarios y de enviar las actualizaciones a los mismos. Lee los archivos de los mapas y los convierte en información útil. También envía peticiones al MySQL para guardar y recuperar información de la base de datos.

El segundo hilo corre la librería MySQL Connector. Esta librería es la que en última instancia gestiona los datos de la base de datos. Cuenta con los siguientes componentes:

- La clase Conexión a la base de datos guarda y recupera información de la base.
- El manejador de consultas convierte peticiones desde el hilo de Kryonet en comandos entendibles por el motor de MySQL.

ISSN: 1390 - 6399

Pruebas de desempeño de la arquitectura.

Para realizar mediciones de la arquitectura, se realizaron modificaciones al juego "Alos" para capturar datos relevantes en archivos externos de texto. La información se importó en Excel, donde se realizaron gráficos con los datos.

La primera medición se realizó para conocer la latencia entre cliente y servidor. Se incluyó en el juego "Alos Online" un lazo que envía mensajes al servidor cada un cuarto de segundo. Se utilizaron ocho terminales variadas (entre móviles y ordenadores), corriendo instancias del juego, enviando mensajes en simultáneo al servidor.

Los mensajes enviados por las ocho terminales cada 250 milisegundos fueron capturados por el servidor y medidos en tiempo de intervalo entre uno y otro.

Para todos los casos, el servidor es un equipo con sistema operativo Windows 7, con 4GB de RAM, ancho de banda de 5,6 Mbps y un procesador de 4 núcleos AMD a 2.6 GHz.

A continuación se realizaron pruebas para medir la estabilidad del servidor. Se programó una bitácora que acumula datos cada vez que la inteligencia artificial realiza los cálculos. Todas las interacciones entre los personajes del mundo virtual y todos los jugadores se realizan cada 500 milisegundos, y luego se envían los resultados a las diferentes terminales según corresponda.

Análisis de Resultados

Mediante un extenso estudio de cada una de las posibilidades planteadas y documentadas, se llegó al resultado de una arquitectura base para un MMORPG. La arquitectura base cuenta con todos los componentes y conexiones necesarias para poder ser implementada en MMORPG. Como prueba de ello, se elaboró un juego apegado fielmente a la arquitectura resultante, logrando un juego funcional capaz de soportar múltiples conexiones y de permitir a varios usuarios interactuar entre sí.

Para comprobar la fiabilidad y usabilidad de la arquitectura, utilizando como herramienta el juego "Alos Online" construido sobre la misma, se realizaron mediciones de desempeño.

Latencia; velocidad de la conexión.

Tras realizar las pruebas de latencia entre el cliente y el servidor, los resultados arrojaron que, aunque existieron picos de hasta cuatro y medio segundos, la mayor concentración de paquetes se recibió entre los 250 ms y los 270 ms como se observa en la Figura 16, con una demora (LAG) promedio de 61 milisegundos entre todas las terminales hacia un mismo servidor, estando este dentro del rango esperado de hasta 120 milisegundos, valor basado en la experiencia con juegos similares.

Carga del servidor; pruebas de estrés.

Al realizar las mediciones de estrés, estas mostraron que la taza de actualización de la inteligencia artificial se mantuvo constante en 500 milisegundos de retraso (el límite impuesto) a medida que las quince terminales diferentes se fueron conectando una a una, como se observa en la Figura 17. Pese a los picos, la taza de actualización no excedió el límite artificial, teniendo un retardo de 0 segundos fuera de lo planeado.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

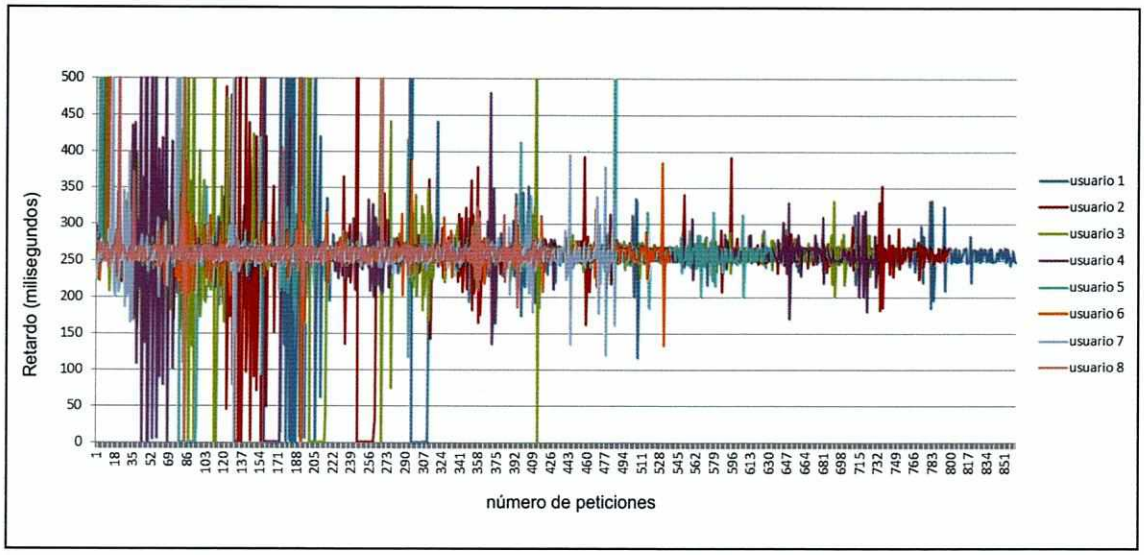

Figura 16. Registro de demora (LAG) de mensajes enviados desde 8 terminales hacia el servidor cada un cuarto de segundo.

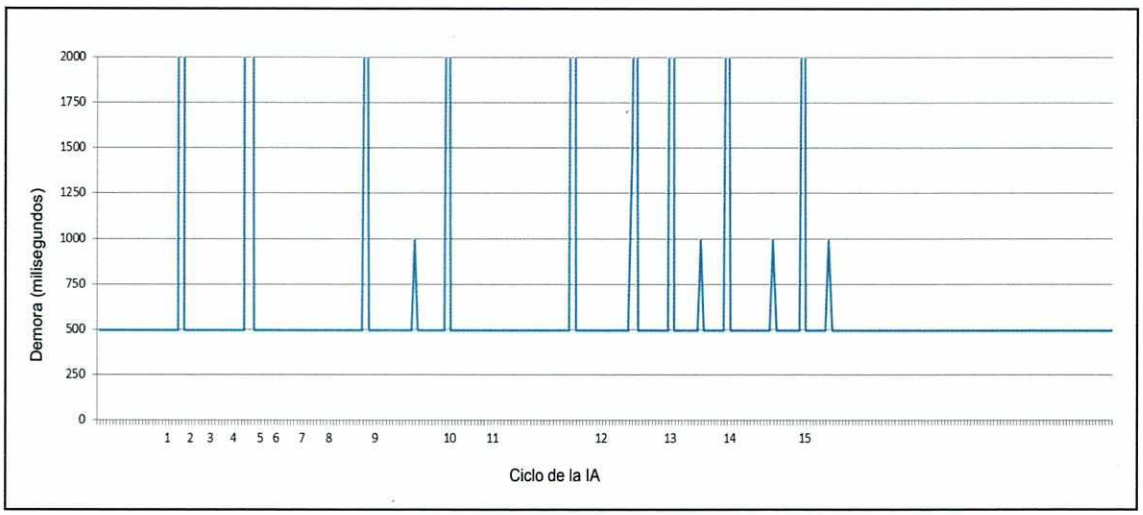

Figura 17. Afectación de la velocidad de la IA a medida que se conectan 15 terminales al servidor.

Simplicidad.

Aunque no se realizaron pruebas de simplicidad, se buscaron estadísticas en la internet. El juego base construido sobre la arquitectura, "Alos Online", llevó alrededor de 4,300 líneas de código en ser construido. Para poner este número en perspectiva, podemos citar a Donkey Kong, el juego de Atari de 1981, con 25,000 líneas de código (Ransom-Wiley, 2009) y a World of Warcraft con 5.5 millones de líneas de código (Sinclair, 2009).

Interconexión

Como por la topología y protocolos elegidos fueron de Cliente-Servidor y TCP más UDP respectivamente, el diseño de todos los clientes conectados a un servidor, garantiza que si un cliente pierde la conexión, el resto no se verían afectadas. Las interconexiones entre usuarios, por diseño, pasan todas por un servidor que las valida y administra. No se realizaron pruebas de P2P por concepto, Cliente-Servidor garantiza la interconexión.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

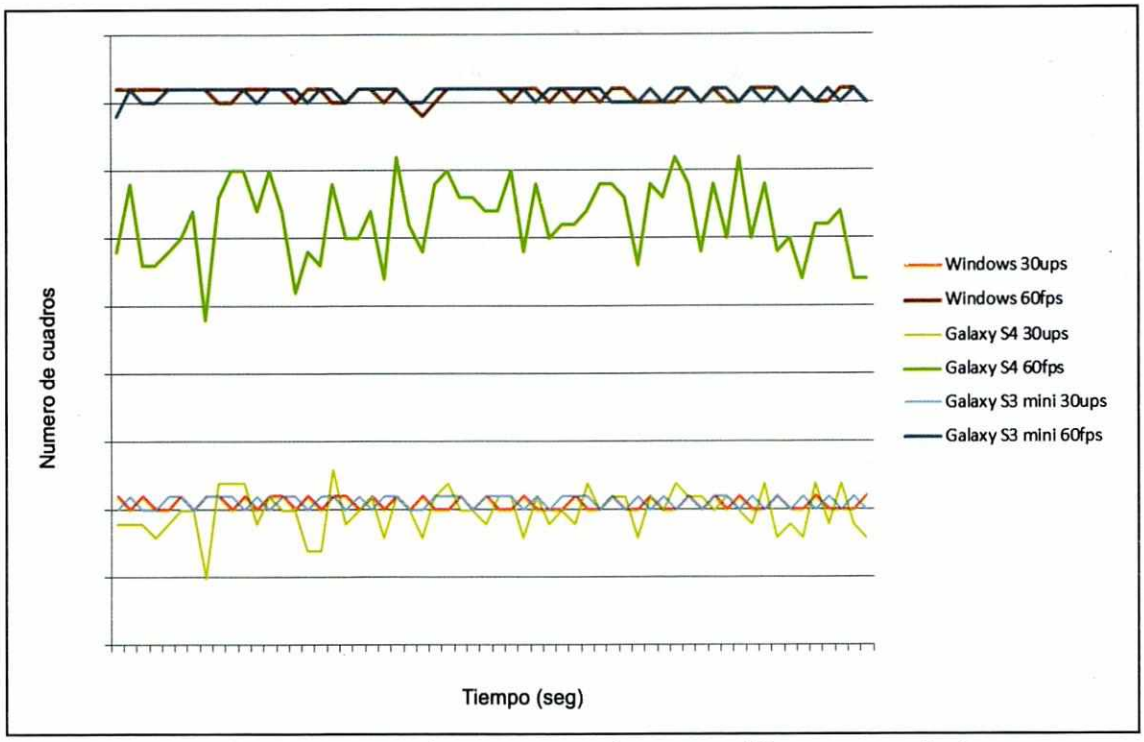

Figura 18. Actualizaciones y cuadros por segundo en tres diferentes terminales

Cuadros por segundo.

Para medir la velocidad del juego, se realizaron pruebas de velocidad del cliente. Se midió la velocidad del juego tanto en cuadros por segundo como en actualizaciones por segundo como se puede observar en la Figura 18.

Se dio a "Alos Online" una taza de actualización de 30 veces en un segundo, y un pintado de pantalla de 60 veces cada segundo. Las mediciones mostraron ser constantes y fiables en la terminal de escritorio, una portátil con procesador i7 a 2.2 GHz y 8Mg de RAM y con sistema operativo Windows 7, teniendo una constante de entre 60 y 61 UPS y entre 30 y 31 FPS. Los resultados de la versión de escritorio fueron los mismos para una terminal móvil de gama baja "Samsung Galaxy S3 mini" con 1GB de RAM y una velocidad de procesador de 1GHz de doble núcleo. La terminal móvil de gama alta "Samsung Galaxy S4", con 2GB de RAM y procesador de 4 núcleos a 1.9GHz, sin embargo, mostro tazas de actualización muy irregulares y por debajo de los parámetros esperados con un UPS entre 25 y 33 milésimas de segundo, y un FPS entre 44 y 56 cuadros por segundo.

Conclusión

Crear un juego de tipo MMORPG no es un trabajo sencillo. Estos requieren de múltiples piezas montadas unas sobre otras y conectadas entre sí para que todo funcione. Crear juegos es una forma de arte (Hussein, 2008), juegos tan complejos como un MMROPG son toda una obra de ingeniería.

Mediante un exhaustivo análisis de varias arquitecturas documentadas se logró esquematizar una de autoría propia, con todos los módulos necesarios para que un juego pueda ser implementado sobre la misma.

Las limitantes para el análisis de la arquitectura se basaron en el costo que representa un servidor dedicado, por lo que las mediciones se efectuaron en un

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

servidor local, y el tiempo de desarrollo para la implementación, ya que lo más común es que proyectos de esta magnitud cuenten con un equipo de trabajo.

Adicionalmente, se suma la limitante de conseguir miles de dispositivos para realizar pruebas tanto de compatibilidad del cliente como pruebas masivas de estabilidad de la conexión. Este tipo de mediciones masivas se logran lanzando versiones "beta" del juego cuando este se encuentre en un estado más avanzado.

Más allá de las limitantes, los objetivos, el de diseño de una arquitectura, implementación de la misma y análisis de la arquitectura utilizando como herramienta el juego creado sobre la misma fueron cumplidos utilizando los recursos disponibles, aunque el segundo no a su total cabalidad:

Las mediciones mostraron baja estabilidad en ciertos dispositivos móviles en cuando a velocidad de actualización y de renderización, problemas que seguramente pueden ser superados al indagar más profundamente el problema. Agregado a esto, quedó pendiente a trabajo futuro pruebas de escalabilidad y de seguridad para cubrir todos los puntos del análisis de la arquitectura.

Sin embargo, y pese a las limitantes se comprobó a cabalidad el correcto funcionamiento de la arquitectura en cuanto a conexiones, con la certeza de que esta puede ser utilizada en proyectos futuros del tipo MMORPG, lo que contribuiría a satisfacer un mercado hambriento de este tipo de juegos.

La arquitectura cumple con su cometido: una estructura para desarrollar juegos MMORPG que, por su género, se desenvuelven en mundos virtuales con posibilidades tan inmensas como lo permite la realidad y la imaginación.

Referencias

Bjørlo, T.C., & Voll, F. (2009). Hybrid Peer-to-Peer Solution for MMORPGs. *NTNU Norwegian University of Science and Technology*, 8.

Chan, H., & Chang, R. (2004). Strifeshadow Fantasy: A Massive Multi-Player Online Game. *Consumer Communications and Networking Conference*, 557-562.

Cidon, I., Rom, R., Gupta, A., & Schuba, C. (1999). Hybrid TCP-UDP Transport for Web Traffic. *Performance, Computing and Communications Conference*, 177-184.

Hendrick, A. (2011, marzo 9). *MMO Tidbits*. Retrieved octubre 5, 2014, from MMO Tidbits: mmotidbits.com.

Huey-Ing Liu, Y.T. L. (2008). DaCAP- A Distributed Anti-Cheating Peer to Peer Architecture for Massive Multiplayer On-line Role Playing Game. *International Symposium on Cluster Computing and the Grid*, 584-589.

Hussein, E.S. K. (2008). A Novel Interactive Computer-Based Game Framework: From Design to Implementation. *International Conference Visualisation*, 123.

Kennedy, R. (2008). Virtual rights? Property in online game objects and characters. *Information & Communications Technology Law*, 95-106.

Kryonet. (s.f.). *GitHub*. Retrieved marzo 15, 2014, from Kryonet: https://github.com/EsotericSoftware/kryonet.

Lee, B.K., Park, C.S., Kim, J.H., Youk, S.-J., & Ryu, K. H. (2008). An Intelligent NPC Framework for Context Awareness in MMORPG. *International Conference on Convergence and Hybrid Information Technology*, 191-194.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

Low Coupling. (n.d.). *Low Coupling*. Recuperado marzo 20, 2014, from Low Coupling: http://lowcoupling.com/post/69270979109/libgdx-handling-inputs-from-keyboard.

Maggiorini, D., Nigro, A., Ripamonti, L. A., & Trubian, M. (2012). Loot Distribution in Massive Online Games: Foreseeing Impacts on the Players Base. *21st International Conference on Computer Communications and Networks (ICCCN)*, 1-5.

Makuch, E. (2014, septiembre 22). *GameSpot*. Recuperado noviembre 30, 2014, de GameSpot: http://www.gamespot.com/articles/steam-reaches-100-million- users-and-3-700-games/1100-6422489.

McGraw, G., & Hoglung, G. (2007). Online Games and Security. *Attack Trends*, 76-79.

MySQL. (s.f.). *MySQL*. Recuperado abril 10, 2014, de MySQL: http://dev.mysql.com/doc/refman/4.1/en/what-is-mysql.html.

Obviam. (2010, agosto 29). *Obviam.net*. Recuperado febrero 20, 2014, de Oviam: Against the Grain: http://obviam.net/index.php/the-android-game-loop/.

Omernick, M. (2004). *Creating the Art of the Game*. New Riders.

OpenGL. (s.f.). *OpenGL*. Recuperado marzo 15, 2014, de http://www.opengl.org/about/#1.

Ransom-Wiley, J. (2009, junio 6). *Engadget*. Retrieved febrero 19, 2015, from Engadget: http://www.engadget.com/ 2009/07/06/donkey-kong-easter-egg-cracked-26-years-late/.

Rouse, M. (2005, agosto 01). *WhatIs.com*. Retrieved abril 09, 2014, from WhatIs.com: http://whatis.techtarget.com/ definition/class-library.

Sinclair, B. (2009, septiembre 17). *Gamespot*. Retrieved febrero 19, 2015, from Gamespot: http://www.gamespot.com/articles/blizzard-outlines-massive-effort-behind-world-of-warcraft/1100-6228615/.

Stallman, R. (2014, marzo 29). *gnu.org*. Recuperado abril 09, 2014, de gnu.org: https://www.gnu.org/philosophy/java-trap.html.

StatisticBrain. (s.f.). *StatisticBrain*. Recuperado diciembre 15, 2014, de StatisticBrain: http://www.statisticbrain.com/top-played-mmorpgs-by-total-number-of-players/.

Suh, s., Kim, S., & Kim, N. (2010). Effectiveness of MMORPG-based instruction in elementary English education in Korea. *Journal of Computer Assisted Learning*, 370-377.

Sung, H. C. (2008, noviembre 8). *IBM*. Recuperado diciembre 12, 2014, de IBM: http://www.ibm.com/developerworks/architecture/library/ar-powerup1/ar-powerup1-pdf.pdf.

SuperData Research Inc. (2015). *SuperData*. Recuperado enero 11, 2015, de SuperData: http://www.superdataresearch.com/market-data/mmo-market/.

Suselbeck, R., Schiele, G., & Becker, C. (2009). Peer-to-Peer Support for Low-Latency Massively Multiplayer Online Games in the Cloud. *8th Annual Workshop on Network and Systems Support for Games (NetGames)*, 1-2.

Suznjevic, M., Dobrijevic, O., & Matijasevic, M. (2009). Hack, Slash, and Chat: A study of players' behavior and communication in MMORPGs. *NetGames '09 Proceedings of the 8th Annual Workshop on Network and Systems Support for Games*.

INVESTIGATIO No. 8, Samborondón, septiembre 2016, pp. 51-70

ISSN: 1390 - 6399

Wu, J., Gong, X., & Zheng, J. (2006). A Model for Massively Multiplayer Role-playing Games System Performance. *IBM Systems Journal (Volume:45 , Issue: 1)*, 45-48.

Wu, Y., Huang, H., & Zhang, D. (2006). Traffic Modeling for Massive Multiplayer On-line Role Playing Game (MMORPG) in GPRS Access Network. *International Conference on Communications, Circuits and Systems Proceedings (Volume:3)*, 1811-1815.

Yan-hui, W., Xia-xia, Y., & Jin, H. (2011). Design and Implementation of the Game Engine based on Android Platform. *International Conference on Internet Technology and Applications (iTAP)*, 1-3.

Yeh, S., Lin, P.J., Liu, H.Y., & Chen, R.M. (2002). The Implementation of Interactive Music System for Massive Multi-player Online Games. *Fourth International Symposium on Multimedia Software Engineering.*, 11-16.

Christian Devetak Cruz

Ingeniero en Sistemas, Universidad Espíritu Santo – Ecuador.

E-mail: christian.com@gmail.com

Iván Francisco Silva Feraud

Máster en Ciencias y Tecnologías de la Computación.
Docente de la Facultad de Sistemas, Universidad Espíritu Santo – Ecuador.

E-mail: ivansilva@uees.edu.ec

ISSN: 1390 - 6399