https://doi.org/10.47300/actasidi-unicyt-2022-58

CONFIGURACIÓN DE UN SERVIDOR MULTIJUGADOR DE MINECRAFT PARA LA CREACIÓN DE ENTORNOS VIRTUALES

Von Lippke Quirola, Darwin Sebastián
Pontificia Universidad Católica del Ecuador sede Ambato
Ambato, Ecuador
Darwin.s.lippke.q@pucesa.edu.ec
ORCID: 0000-0002-3355-227X

Pailiacho Mena, Verónica Maribel
Pontificia Universidad Católica del Ecuador – Ambato
Ambato, Ecuador
vpailiacho@pucesa.edu.ec
ORCID: 0000-0002-8394-3148

Garcés Freire, Enrique Xavier
Pontificia Universidad Católica del Ecuador – Ambato
Ambato, Ecuador
egarces@pucesa.edu.ec
ORCID: 0000-0002-5566-6825

RESUMEN

La globalización y sociedades del conocimiento empiezan a aprovechar las herramientas de la industria de la tecnología. Existen trabajos académicos que utilizan a los videojuegos como un medio de llegar al conocimiento al utilizar ambientes digitales que estos ofrecen, sin embargo, no todos presentan destrezas profundas de la utilización de estos. El videojuego Minecraft se utiliza en diferentes estudios realizados de temas variados por lo que crear un servidor multijugador es un paso implícito que se realiza y en la mayoría de los casos no se documenta. Con el propósito de contribuir con la educación que utiliza entornos virtuales de juegos comerciales, el presente trabajo tiene la finalidad de aportar con una configuración eficiente para el desarrollo de escenarios virtuales en el juego de Minecraft. Como resultado se presenta una guía para entender conceptos básicos, funcionamiento general, optimización, creación de un servidor y la forma de desarrollar entornos virtuales optimizados.

Palabras clave: educación superior, entornos virtuales, Minecraft, videojuegos

ABSTRACT

Globalization and knowledge societies are beginning to take advantage of the tools of the technology industry. There are academic works that used video games as a means of reaching knowledge by using digital environments that they offer, however, not all of them present deep skills in the use of these. The Minecraft video game is used in different studies on various topics, so creating a multiplayer server is an implicit step that is carried out and in most cases is not documented. With the purpose of contributing to the education that uses virtual environments of commercial games, the present work has the purpose of contributing with an efficient

configuration for the development of virtual scenarios in the Minecraft game. As a result, a guide is presented to understand basic concepts, general operation, optimization, creation of a server and how to develop optimized virtual environments.
Keywords: higher education, Minecraft, video games, virtual environments

1. INTRODUCCIÓN

En los últimos años la industria de tecnología empieza a cambiar la forma en que las personas se comunican, relacionan, trabajan y entretienen. El autor Lindo (2015), manifiesta que la utilización de los videojuegos fortalece el desarrollo de habilidades, competencias y valores necesarios para un comportamiento proactivo. La utilización de los videojuegos puede facilitar el proceso de aprendizaje, desarrollo de habilidades y conocimiento en múltiples áreas. La nueva era digital pone nuevas formas de interrelación para aprovechar la narrativa intermedial que estas brindan (Sanchis, 2020). Según & Alarcón (2020), la utilización de videojuegos como forma de digitalización ha favorecido en el desarrollo de la participación de estudiantes de manera inusitada. Esto muestra la utilización de escenarios digitales para llamar la atención y dar experiencias nuevas al usuario a comparación de hacerlo con métodos tradicionales.

Minecraft (https://Minecraft.net/) es usado para el desarrollo de entornos digitales, (2015), explica que Minecraft es un juego de mundo abierto en donde se puede construir cualquier cosa y es muy fácil de manejar. Según Valldecabres (2016), este videojuego ha tenido un gran auge desde su lanzamiento por su simplicidad y las posibilidades que ofrece al ser de mundo abierto. Ponce & Alarcón (2020), manifiestan que al utilizar el motor de búsqueda académica de Google se presencia una gran cantidad de trabajos académicos sobre Minecraft y sus distintas aplicaciones en la educación en los que se muestran varias perspectivas de Minecraft en las que los autores concluyen que Minecraft es un videojuego con características sencillas que brinda libertad a los usuarios al momento de jugar y construir un mundo abierto, y el manejo de mecánicas simples ha hecho que se popularice convirtiéndose en una herramienta de apoyo para la educación.

Existe una versión académica de Minecraft (https://education.minecraft.net/), esta versión presenta entornos digitales diseñados específicamente para el aprendizaje. Sin embargo, estos entornos ya están creados con lecciones predefinidas y no son modificables, por lo que no se puede crear entornos personalizados que se adapten a necesidades específicas.

2. MARCO CONCEPTUAL

MINECRAFT

Minecraft es un videojuego del tipo supervivencia en el cual los jugadores pueden moverse en un mundo abierto, en este se puede construir objetos y entornos mediante cubos con texturas tridimensionales (Lindo, 2015).

SERVIDOR DE MINECRAFT

Es una instancia en la cual se ejecuta una versión modificada del videojuego Minecraft. Mojan, la desarrolladora de Minecraft proporciona una versión oficial del videojuego especializada para servidores. Según Valldecabres (2016) las versiones oficiales de servidor de Minecraft se llaman *vanilla,* la cual es una versión sin modificaciones, mientras que a las versiones no oficiales se los conoce como *mods*. Las versiones *vanilla* no están optimizadas por lo que se opta por utilizar las versiones modificadas las cual permite la incorporación de complementos.

COMPLEMENTOS

Para Valldecabres (2016) los complementos son pequeños programas que aumentan o modifican las funcionalidades básicas de un servidor de Minecraft. Estos complementos pueden facilitar la administración de un servidor de Minecraft como proporcionar sistemas de gestión de usuarios, optimización de rendimiento o sistemas de interfaces gráficas.

ARQUITECTURA CLIENTE/SERVIDOR

Modelo que consta de dos actores, el primero es el cliente, el cual realiza peticiones a un programa para solicitar recursos y el servidor el cual responde y acepta esas conexiones y las satisface al proveer de recursos o servicios (Lizama, 2016). El servidor comienza sus servicios antes de que se realice las solicitudes por parte de los clientes mientras que ellos realizan la petición y termina el proceso.

MULTIUSUARIO

Un sistema multiusuario es aquel que permite trabajar a varios usuarios en un mismo ordenador (Molina, 2003). Este sistema permite a varios jugadores de Minecraft interactuar entre sí en una o varias instancias.

3. MATERIALES Y MÉTODOS

La presente investigación busca responder a la pregunta: Cómo obtener una configuración de un servidor multijugador de Minecraft eficiente?; para encontrar la respuesta a esta pregunta se planteó hacer una búsqueda en base a los siguientes términos claves: Minecraft, Estándar, Optimización, Creación, Servidor, éstos términos de búsqueda se aplicaron en los motores de: Google Académico, Microsoft Academic, para encontrar documentación académica que permita identificar información valida, de esta primera búsqueda se encuentra información que dirige a sitios oficiales de creación y optimización de servidores de Minecraft, sitios como: Página oficial de Minecraft que proporciona una copia del juego enfocada a servidor. (https://www.minecraft.net/es-es/download/server) y PaperMC (https://papermc.io), y se hizo una instalación y configuración.

Con la información encontrada se procede a hacer el análisis y presentación de los hallazgos, en el siguiente punto.

4. RESULTADOS Y DISCUSIÓN

Para realizar la experimentación se utilizó:

- Servidor / ordenador
- Sistema operativo Debian 10.13
- MariaDB 10.11
- Waterfall construcción 504
- Paper 1.19.2
- JDE Java 17

La ejecución e implementación del trabajo de investigación consta de cuatro partes:

1. Infraestructura del servidor de Minecraft
2. Organización de un servidor de Minecraft
3. Optimización de las instancias
4. Creación de los entornos virtuales

4.1. INFRAESTRUCTURA DEL SERVIDOR DE MINECRAFT

Fase 1. Los entornos comerciales masivos multiusuario como Minecraft funcionan con arquitecturas cliente/servidor y cliente/servidor distribuido (Engelbrecht & Schiele, 2014). Se puede crear espacios virtuales de Minecraft en servidores locales y crear una intranet para el acceso a ellos. Sin embargo, se alquiló una máquina privada a una empresa de centro de datos para tener un espacio de trabajo disponible permanentemente en el cual poder configurar los entornos multijugador. Para la selección del sistema operativo se escogió Debian, el cual es una

distribución libre de Linux, la selección de este software radica en su robustes que ofrece al momento de crear un servidor de aplicaciones y su ligereza puesto que su instalación no cuenta con herramientas extra no deseadas.
Para poder realizar las instalaciones y configuraciones necesarias se conectó por medio de una terminal por SSH, se realizó esta conexión porque se alquila una máquina situada en otro país.
Se instaló MariaDB el cual es un gestor de base de datos estructurada necesaria para el almacenaje de datos generados por los complementos de Minecraft.
También se instaló la versión 17 de Java la cual es requerida para ejecutar la última distribución de Minecraft (1.19.2).
Para que se pueda realizar la conexión cliente al servidor se necesitó habilitar los siguientes puertos:

- **Puerto 25565:** Este puerto debe aceptar únicamente conexiones con el protocolo TCP/IP, va destinado a los usuarios que se conecten con la versión de Minecraft Java Edition.
- **Puerto 19132:** Este puerto va destinado a las conexiones realizadas desde dispositivos móviles y consolas de videojuegos que utilizan el protocolo UDP.
- **Puerto 18012:** Este puerto se destina a conexiones de SSH y SFTP. Se puede usar cualquier número, pero se recomiendo no dejar el puerto predeterminado para estos protocolos (puerto 22) por seguridad.

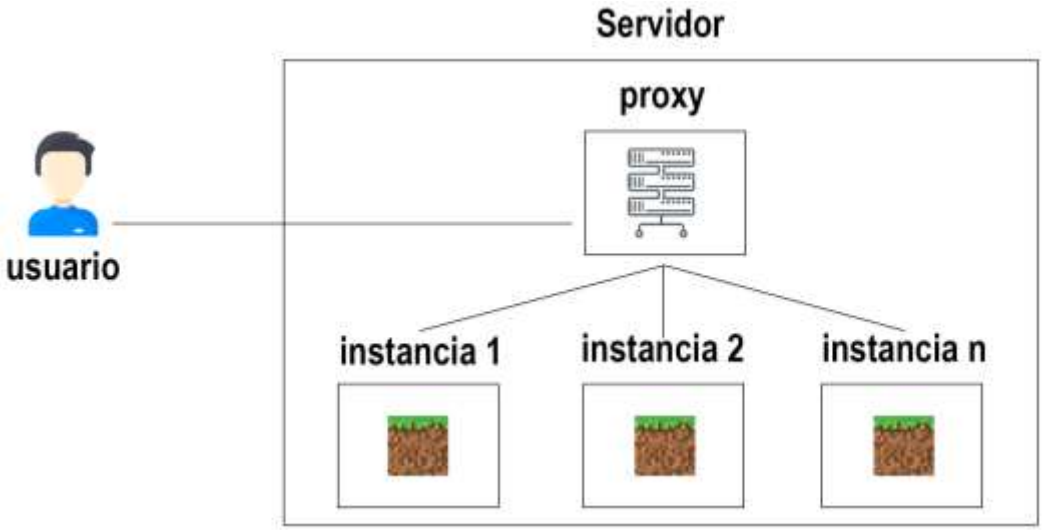
El resto de los puertos es recomendable cerrarlos para no tener problemas de seguridad.

Fase 2. Para la siguiente fase se creó un nuevo usuario para el sistema operativo y un directorio, este usuario tiene autorizaciones para gestionar únicamente el nuevo directorio creado. Para la creación de los entornos y suma de complementos se debe realizar con este usuario para evitar problemas de permisos. Conjuntamente se creó una base de datos y un usuario para ella.

4.2. ORGANIZACIÓN DE UN SERVIDOR DE MINECRAFT

La arquitectura que se utilizó fue cliente/servidor distribuido, en la que las instancias de Minecraft están trabajando de manera paralela y todas estas serían nodos formando un clúster (Engelbrecht & Schiele, 2014).

Figura 1.
Distribución del servidor de Minecraft en forma de clúster.



Un servidor de Minecraft distribuido contiene un proxy el cual permite la migración de jugadores entre nodos del clúster. Para ello se utiliza un software de proxy exclusivo para Minecraft llamado Waterfall (https://papermc.io/downloads#Waterfall) el cual permite realizar el puente entre instancias. Este es una versión mejorada y estable del proxy BungeeCord. Existen otros softwares que realizan la función de proxy sin embargo este es uno de los más estables y compatibles con la mayoría de los complementos.
Para que funcione una instancia se debe utilizar el software PaperMC(https://papermc.io/downloads#Paper-1.19), el cual es una versión para servidor de alto rendimiento que tiene como objetivo corregir inconsistentes mecánicas de juego original. Existen bifurcaciones de este programa los cuales agregan y mejoran características concretas, sin embargo, estas versiones son menos estables o tienen un soporte bastante bajo.
Se crearon dos carpetas dentro de la principal y se agregó los softwares PaperMC, para las instancias, y Waterfall para el proxy. Debido a que Debian no posee una interfaz gráfica por defecto, se multiplexó la terminar para que el proxy y cada instancia pueda ir en pantallas individuales para que sus procesos no se cierren. Para poder arrancar los servidores se necesita agregar a cada carpeta un ejecutable que los inicie. Para dicha acción se creó un archivo start.sh con la siguiente línea de comando:

- Para el proxy: *java -Xms1g -Xmx1g -jar waterfall.jar*
- Para la instancia: *java -Xms2g -Xmx2g -jar server.jar*

Al arrancar las instancias por primera vez se generó los archivos y carpetas correspondientes al juego, pero de forma incompleta hasta que se acepta el acuerdo de licencia de usuario final de Minecraft (EULA), el cual, al aceptarlo, continuó con la descarga del resto de archivos.

4.3. OPTIMIZACIÓN DE LAS INSTANCIAS Y ASIGNACIÓN DE PUERTOS

Al crear un nuevo servidor de Minecraft los archivos generados vienen con una configuración predeterminada que al no ser cambiada será ineficiente. *Bukkit.yml, spigot.yml, paper.yml y server.properties* son archivos que contienen parámetros que determinan el rendimiento del servidor. A continuación, se muestra la configuración optima con el cual se mejoró el rendimiento del servidor. Estos archivos contienen parámetros definidos los cuales no todos se cambian.

Bukkit.yml

```
spawn-limits:
  monsters: 20
  animals: 5
  water-animals: 2
  water-ambient: 2
  water-underground-creature: 3
  axolotls: 3
  ambient: 1
chunk-gc:
  period-in-ticks: 400
ticks-per:
  monster-spawns: 10
  animal-spawns: 400
  water-spawns: 400
  water-ambient-spawns: 400
  water-underground-creature-spawns: 400
  axolotl-spawns: 400
```

```
  ambient spawns: 400
```

Spigot.yml

```
bungeecord: true
item-despawn-rate: 6000
arrow-despawn-rate: 300
nerf-spawner-mobs: true
mob-spawn-range: 2
merge-radius:
  item: 3.5
  exp: 4.0
entity-activation-range:
  animals: 16
  monsters: 24
  raiders: 48
  misc: 8
  water: 8
  villagers: 16
  flying-monsters: 48
  villagers-work-immunity-after: 100
  villagers-work-immunity-for: 20
  villagers-active-for-panic: true
  tick-inactive-villagers: false
  wake-up-inactive:
    animals-max-per-tick: 2
    animals-every: 1200
    animals-for: 40
    monsters-max-per-tick: 4
    monsters-every: 400
    monsters-for: 60
    villagers-max-per-tick: 1
    villagers-every: 600
    villagers-for: 20
    flying-monsters-max-per-tick: 1
    flying-monsters-every: 200
    flying-monsters-for: 60
tick-inactive-villagers: false
entity-tracking-range:
  players: 48
  animals: 48
  monsters: 48
  misc: 32
  other: 64
hopper-transfer: 8
hopper-check: 8
```

Paper.yml

```
environment:
  optimize-explosions: true
  nether-ceiling-void-damage-height: 127
```

chunks:
max-auto-save-chunks-per-tick: 8
prevent-moving-into-unloaded-chunks: true
delay-chunk-unloads-by: 10s
entities:
spawning:
per-player-mob-spawns: true
non-player-arrow-despawn-rate: 20
creative-arrow-despawn-rate: 20
behavior:
spawner-nerfed-mobs-should-jump: true
disable-chest-cat-detection: true
collisions:
max-entity-collisions: 2
fix-climbing-bypassing-cramming-rule: true
misc:
update-pathfinding-on-block-update: false
redstone-implementation: ALTERNATE_CURRENT
tick-rates:
behavior:
villager:
validatenearbypoi: -1
container-update: 1
grass-spread: 4
mob-spawner: 2
sensor:
villager:
nearestbedsensor: 80
nearestlivingentitysensor: 40
playersensor: 40
secondarypoisensor: 80
villagerbabiessensor: 40 armor-stands:
do-collision-entity-lookups: false
tick: false
treasure-maps:
enabled: false
find-already-discovered:
loot-tables: 'true'
villager-trade: true
hopper:
disable-move-event: true
Server.properties:
network-compression-threshold=1
simulation-distance=4
view-distance=7

***Config.yml* del proxy**
network_compression_threshold: 128

Después de realizar la optimización se asignó los puertos el archivo *config.yml* del servidor proxy de la siguiente manera:

```
servers:
  instancia1:
    motd: 'server'
    address: localhost:18018
- query_port: 25577
  priorities:
  -    instancia1
  host: 0.0.0.0:25565
  force_default_server: true
```

Y en el archivo *server.properties* de la instancia se colocó *server-port=18018*, mismo número que se definió anteriormente.

4.4. CREACIÓN DE LOS ENTORNOS VIRTUALES

Para finalizar esta investigación se creó un entorno virtual vacío, ideal para comenzar a construir y dar forma a proyectos concretos. Se puede desarrollar entornos de cualquier tema como por ejemplo los escenarios de una novela literaria que presenta pocas ilustraciones o directamente carece de ellas, recrear sucesos históricos y culturales, o simplemente crear un espacio de convivencia.

Figura 2.

Collage de diferentes entornos digitales en minecraft.

Una vez construido un entorno plantilla para tener las mismas configuraciones y complementos se realizó las pruebas de rendimiento con los siguientes componentes:

- Ordenador: AMD Ryzen™ 5 3600
- CPU: 6 núcleos / 12 hilos @ 3.6 GHz
- Generación: Matisse (Zen 2)
- RAM: 64 GB DDR4 RAM
- Almacenamiento: 2 x 512 GB NVMe SSD
- Sistema operativo: Debian 10
- Base de datos: MaríaDB 8.1

Para medir el rendimiento general de un servidor de Minecraft se utiliza los *ticks per second* (TPS), el cual determina si la instancia está bien o tiene problema de rendimiento. Esta información es recopilada por el mismo servidor y presentada mediante una gráfica.
El número máximo de TPS es de 20 siendo esta la cantidad ideal para un entorno y de 19 a 17 TPS es aceptable. El rendimiento sigue bajando hasta 14 TPS siendo el límite jugable, pero teniendo problemas de latencia. Debajo de este número el servidor tiene errores entre complementos, realiza desconexiones con la base de datos y se torna inestable hasta tener un cierre.
A cada instancia se le asignó 6 GB de RAM para ejecutar un entorno con 42 complementos. Para la primera instancia se utilizó los parámetros de configuración predeterminados, tuvo 17 TPS con cinco jugadores simultáneos y con un pico de 15 jugadores provocó un colapso al servidor. Para la segunda instancia se aplicó la optimización y pudo mantener a 50 jugadores simultáneos con 20 TPS, se aumentó los usuarios hasta llegar a un límite de 65 jugadores sin pérdida significativa de rendimiento. A partir del jugador 66 empezó a tener el servidor 18 TPS y con 90 jugadores colapsó la instancia.

5. CONCLUSIONES

Una vez terminadas las pruebas de rendimiento con un total de 14 servidores testeados durante dos años se puede manifestar que la utilización de esta configuración mejoró el rendimiento de las instancias de 10 a 12 veces más, siendo 50 jugadores simultáneos la cantidad optima de usuarios permitidos sin afectar el rendimiento. Se pudo llegar al objetivo de crear una configuración optimizada para crear entornos virtuales Minecraft que en esencia será útil para la creación de estas para nuevas líneas de investigación. La documentación que se presentó es una introducción el entendimiento del uso, configuración y administración de un entorno virtual mediante la adaptación de un videojuego comercial. A su vez se presenta una nueva visión más cuidada de la utilización de estos medios para el fortalecimiento de conocimientos en múltiples disciplinas.
Además, ser estudiante de ingeniería y crear un servidor de Minecraft ha aportado al aprendizaje y fortalecimiento de varios conceptos revisados en clases para la implementación del servidor como son sistemas operativos, bases de datos, redes, y programación.

REFERENCIAS

Molina, J. (2007). Sistemas Operativos en Entornos monousuario y multiusuario. Windows 2003 Server y Linux. Vision Libros. https://books.google.hn/books?id=yECVWwb4kL0C&printsec=copyright&hl=es#v=onepage&q&f=false

Engelbrecht, H. A., & Schiele, G. (2014). Transforming Minecraft into a research platform. In *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)* (pp. 257-262). IEEE. http://doi.org/10.1109/CCNC.2014.6866580

Sanchis, M. D. L. (2020). Antonio J. Gil González & Pedro Javier Pardo (eds.). Adaptación 2.0. Estudios comparados sobre intermedialidad. Revista 2i: Estudos De Identidade E Intermedialidade, 2(1), 189–192. https://doi.org/10.21814/2i.2697

Lindo, C., Sanz, P., De Benito, J., & Galindo, J. (2015). Aprendizaje del Lean Manufacturing mediante Minecraft: aplicación a la herramienta 5S. RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação, (16), 60-75. https://doi.org/10.17013/risti.e4.66-78

Lizama, O., Kindley, G., Morales, J. J., & Gonzales, A. (2016). Redes de Computadores: Arquitectura Cliente-Servidor. Universidad Tecnica Federico Santa María, 1-8. http://profesores.elo.utfsm.cl/~agv/elo322/1s16/projects/reports/Proyecto%20Cliente%20-%20Servidor.pdf

Ponce, R., & Alarcón, L. (2018). Videojuego Minecraft como recurso para la alfabetización académica en la educación superior. Actualidades Investigativas en Educación, 18(3), 664-680. http://dx.doi.org/10.15517/aie.v18i3.34382

Valldecabres, J. (2016). *CraftCosta: Creación de un plug-in MMORPG para servidores de Minecraft* (Doctoral dissertation, Universitat Politècnica de València). https://riunet.upv.es/handle/10251/70973

i

[i] Los autores del trabajo autorizan a la Universidad Internacional de Ciencia y Tecnología (UNICyT) a publicar este resumen en extenso en las Actas del Congreso IDI-UNICyT 2022 en Acceso Abierto (Open Access) en formato digital (PDF) e integrarlos en diversas plataformas online bajo la licencia CC: Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0) https://creativecom mons.org/licenses/by-nc-sa/4.0/.

La Universidad Internacional de Ciencia y Tecnología y los miembros del Comité Organizador del Congreso IDI-UNICyT 2022 no son responsables del contenido ni de las implicaciones de lo expresado en este artículo.