International Journal of Information Science and Technological Applications-UAS

IJISTA

Junio-Noviembre 2026 Vol. II Núm. II

UNIVERSIDAD AUTÓNOMA DE SINALOA

ISSN: 3122-4474

UNIVERSIDAD AUTÓNOMA DE SINALOA

International Journal of Information Science and Technological Applications-UAS

IJISTA

REVISTA

Facultad de Informática Culiacán

Culiacán de Rosales, Sinaloa, México.

https://revistas.uas.edu.mx/index.php/IJ ISTA/index

DIRECTORIO INSTITUCIONAL

Dr. Jesús Madueña Molina
Rector

Dra. Nidia Yuniba Brunn Corona
Secretaria General

Dr. Alfonso Mercado Gómez
Director de Servicios Escolares

Dr. Sergio Mario Arredondo Salas
Secretario Académico Universitario

Dr. Wenseslao Plata Rocha
Vicerrector de Unidad Regional Centro

Dra. Marcela de Jesús Vergara Jiménez
Directora General Investigación y Posgrado

Dr. Joel Cuadras Urías
Director General del Sistema Bibliotecario

Dr. Zeus Del Valle Castillo Nájera
Director de la Facultad de Informática Culiacán

Dra. Xiomara Penélope Zaldívar Colado
Editora en Jefe de la Revista IJISTA-UAS

Comité Editorial

Comité Editorial

Comité Científico

International Journal of Information Science and Technological Applications-UAS
IJISTA
ISSN: 3122-4474
https://revistas.uas.edu.mx/index.php/IJISTA

Junio 2026
Vol. II.
Número. II

Cintillo Legal

Google Scholar

ROAD DIRECTORY OF OPEN ACCESS SCHOLARLY RESOURCES

CREATIVE COMMONS BY NC SA

LACLI

CONTENIDO

Editorial

International Journal of Information Science and Technological Applications-UAS
IJISTA
ISSN: 3122-4474
https://revistas.uas.edu.mx/index.php/IJISTA

Junio 2026
Vol. II.
Número. II

Editorial

Carta del Editor

Estimadas lectoras y lectores:

Es un honor presentar el Volumen II, Número II de la revista científica multidisciplinaria, International Journal of Information Science and Technological Applications - UAS IJISTA, publicación de la Universidad Autónoma de Sinaloa dedicada a difundir investigación original en ciencias de la información y sus aplicaciones tecnológicas.

En este número reunimos cinco trabajos que, desde ángulos distintos, ilustran la vitalidad de nuestra comunidad académica: el desarrollo independiente de videojuegos, la ciberseguridad en infraestructuras de red, la automatización de procesos organizacionales mediante inteligencia artificial, el monitoreo proactivo de redes locales en entornos educativos, y la computación paralela aplicada a la simulación numérica.

Abrimos el número con “Análisis Cuantitativo de la Eficiencia en la Paralelización OpenMP de la Ecuación de Difusión: Un Estudio Experimental de Escalabilidad y Umbrales Críticos”, de José Joel Ruelas Leal, Salvador Meza Aguilar y Francisco César Delgado Nieblas, trabajo desarrollado en Facultades de Informática y de Ciencias Físico-Matemáticas. Mediante un diseño factorial completo con mallas de Nx=200 a Nx=20,000 y configuraciones de uno a ocho hilos, los autores identifican un umbral crítico en Nx=20,000 a partir del cual la paralelización resulta favorable, y ofrecen directrices basadas en evidencia para el diseño de implementaciones paralelas eficientes en computación científica.

El segundo trabajo, “Implementación de Ciberseguridad en Redes mediante MikroTik y Servicios en Debian”, de Denisse Carolina Navarro García, Sayda Karely Carmona Medina y Henry Osuna Boltor, también de la Facultad de Informática Mazatlán, documenta la construcción de un entorno de red seguro que combina reglas de firewall en RouterOS, servicios desplegados sobre Debian y cifrado HTTPS mediante Certbot.

Los autores demuestran cómo un entorno inicialmente vulnerable puede transformarse, mediante configuraciones accesibles, en una infraestructura con control de acceso y monitoreo efectivos.

En tercer lugar, Eduardo Montes de Oca Zatarain, Carlos Tirado Velázquez, Erik Watson Rosales y Abraham Páez Guerra presentan "Uso de Inteligencia Artificial en la Automatización de Procesos Organizacionales". A partir de un enfoque mixto que combina revisión documental y simulación teórica, el estudio proyecta mejoras de eficiencia operativa de entre 50% y 80%, y reducciones de errores humanos de hasta 90%, reafirmando el papel del aprendizaje automático como pilar de la transformación digital en áreas como recursos humanos y logística, sin dejar de señalar la necesidad de validación empírica y de atender los desafíos éticos de transparencia y sesgo algorítmico.

El cuarto artículo, "FlowTrix: Arquitectura Persistente para el Monitoreo Proactivo y Detección de Vulnerabilidades en Redes Locales", de José Carlos Castillo Padilla, José Gerardo Sánchez Rodríguez y José David Santana Alaniz, propone una plataforma de monitoreo compuesta por un agente en C#/.NET 8, un backend REST en PHP y un panel de clasificación dinámica de riesgo por host. Validada en 40 a 50 estaciones de trabajo del Laboratorio de Cómputo de la Facultad de Informática Mazatlán, y sometida a pruebas de carga con 100 agentes simultáneos, FlowTrix alcanzó una integridad de registros del 99.7% y una precisión de detección de puertos no autorizados del 95%, con un consumo de CPU inferior al 1%, evidenciando que la auditoría proactiva de endpoints es viable en entornos institucionales de pequeña escala.

Cerramos el número con "Herramientas gratuitas para desarrolladores principiantes de videojuegos", de Emmet Crespo-Rojas, Jose Armando Villalobos-Puga, Gerson Gibran Ramírez-Holguín, Félix David Durán-Hernández y Luis David Sánchez-Fragoso, de la Facultad de Informática Mazatlán. El artículo caracteriza un ecosistema de nueve herramientas de código abierto distribuidas en cinco categorías de producción —motores de videojuego, arte 2D y 3D, audio y assets— y muestra que, bajo licencias MIT y GNU GPL, el tiempo y el esfuerzo del desarrollador se convierten en los principales recursos para llevar un proyecto de principio a fin sin barreras económicas de licenciamiento.

En conjunto, estos cinco trabajos reflejan el doble compromiso de IJISTA: por un lado, con la investigación aplicada que resuelve problemas concretos de infraestructura, seguridad y automatización; por otro, con el rigor metodológico que sustenta la generación de conocimiento científico verificable. Agradecemos a las autoras y autores por su dedicación,

así como a las personas que participaron como revisoras y revisores, cuyo trabajo anónimo y comprometido garantiza la calidad editorial de cada número.

Invitamos a la comunidad académica de la Universidad Autónoma de Sinaloa y de instituciones afines a continuar enviando sus contribuciones, y agradecemos a nuestros lectores por acompañarnos en este nuevo número.

Atentamente,

Xiomara Penélope Zaldivar Colado
Editora en Jefe

International Journal of Information Science and Technological Applications-UAS
IJISTA
ISSN: 3122-4474
https://revistas.uas.edu.mx/index.php/IJISTA

Junio 2026
Vol. II.
Número. II

Análisis Cuantitativo de la Eficiencia en la Paralelización OpenMP de la Ecuación de Difusión: Un Estudio Experimental de Escalabilidad y Umbrales Críticos

Quantitative Analysis of OpenMP Parallelization Efficiency for the Diffusion Equation: An Experimental Study of Scalability and Critical Thresholds

José Joel Ruelas Leal[1], Salvador Meza Aguilar[2], Francisco Cesar Delgado Nieblas[1]

[1]Facultad de informática, Universidad Autónoma de Sinaloa, Mexico.

https://orcid.org/0009-0002-0940-4530
jj.ruelas20@info.uas.edu.mx

https://orcid.org/0000-0002-2622-3689
smeza@uas.edu.mx

Autor de Correspondencia: Francisco C. Delgado Nieblas
francisco.delgado@info.uas.edu.mx
https://orcid.org/0009-0007-5593-3341

Recibido: abril 2026
Publicado: junio 2026

IJISTA, Vol. II, Núm. I, junio 2026,
ISSN: 3122-4474

Resumen:

La simulación numérica de ecuaciones diferenciales parciales mediante métodos de diferencias finitas presenta demandas computacionales que crecen rápidamente con la resolución espacial. Este trabajo analiza cuantitativamente la eficiencia de la paralelización OpenMP para la ecuación de difusión unidimensional resuelta con el esquema FTCS. Se implementó un diseño experimental factorial completo con tamaños de malla desde Nx=200 hasta Nx=20000 y configuraciones de 1 a 8 hilos. Los resultados demuestran la existencia de un umbral crítico en Nx=20000, donde se alcanza el primer speedup superior a la unidad (1.05× con 4 hilos). Para problemas pequeños (Nx=200), la paralelización resulta contraproducente debido al overhead asociado a sincronización y administración de hilos. La eficiencia paralela disminuye conforme aumenta el número de hilos, aunque mejora progresivamente para problemas con mayor carga computacional. Mediante la Ley de Amdahl se estimó una fracción paralelizable elevada, mientras que el análisis experimental evidenció la influencia significativa del overhead práctico sobre el rendimiento observado. La precisión numérica se conserva entre versiones serial y paralela hasta el orden de 10^{-14}. Este trabajo proporciona directrices prácticas basadas en evidencia experimental para implementaciones paralelas eficientes en computación científica.

Palabras Clave:

OpenMP, ecuación de difusión, paralelización, speedup, eficiencia paralela, métodos numéricos.

Abstract:

Numerical simulation of partial differential equations using finite difference methods presents computational demands that grow rapidly with spatial resolution. This work quantitatively analyzes the efficiency of OpenMP parallelization for the one-dimensional diffusion equation solved using the FTCS scheme. A full factorial experimental design was implemented with mesh sizes ranging from Nx=200 and thread configurations from 1 to 8 threads. Results demonstrate the existence of a critical threshold at Nx=20000, where the first speedup greater than unity is achieved (1.05× using 4 threads). For small problems (Nx=200), parallelization becomes counterproductive due to synchronization and thread-management overhead. Parallel efficiency decreases as the number of threads increases, although performance progressively improves for larger computational workloads. Using Amdahl's Law, a high parallelizable fraction was estimated, while experimental analysis revealed the significant impact of practical overhead on the observed performance. Numerical precision is preserved between serial and parallel implementations up to the order of 10^{-14}. This work provides evidence-based practical guidelines for efficient parallel implementations in scientific computing.

Keywords:

OpenMP, diffusion equation, parallelization, speedup, parallel efficiency, numerical methods.

IJISTA, Vol. II, Núm. I, junio 2026, ISSN 3122-4474

1. Introducción

La ecuación de difusión constituye uno de los modelos fundamentales de la física matemática y aparece en una amplia variedad de aplicaciones científicas y tecnológicas, incluyendo transferencia de calor, dispersión de contaminantes, transporte de masa y modelos financieros [1] [2]. Su resolución numérica mediante métodos de diferencias finitas continúa siendo relevante debido a la necesidad de simular sistemas con alta resolución espacial y temporal. Entre los esquemas explícitos más utilizados se encuentra el método FTCS (Forward Time Centered Space), ampliamente estudiado por su simplicidad de implementación y sus propiedades de convergencia [2].

El incremento continuo en la resolución de las simulaciones numéricas ha provocado un crecimiento considerable en los requerimientos computacionales. En particular, la condición de estabilidad asociada a esquemas explícitos como FTCS obliga a reducir el paso temporal conforme disminuye el espaciamiento espacial, incrementando significativamente el número total de operaciones necesarias para completar una simulación [3], [4]. Como consecuencia, incluso problemas unidimensionales pueden presentar tiempos de ejecución elevados cuando se emplean mallas suficientemente refinadas.

En las últimas décadas, la evolución de las arquitecturas de hardware ha favorecido el uso de procesadores multinúcleo como estrategia principal para incrementar el rendimiento computacional. En este contexto, OpenMP se ha consolidado como uno de los estándares más utilizados para programación paralela en sistemas de memoria compartida debido a su portabilidad y facilidad de integración en códigos científicos existentes [5], [6]. Su modelo basado en directivas permite paralelizar regiones específicas del código sin modificar completamente la estructura original del programa.

A pesar de las ventajas de la programación paralela, la paralelización no garantiza automáticamente mejoras de rendimiento. Factores como el overhead de creación y sincronización de hilos, la contención de memoria compartida y la fracción serial del código pueden limitar considerablemente el speedup alcanzable [7]. En algunos casos, especialmente para problemas pequeños, la versión paralela puede incluso presentar tiempos de ejecución mayores que su contraparte serial.

Aunque existe una amplia literatura sobre métodos numéricos y programación paralela, todavía son limitados los estudios experimentales que cuantifican sistemáticamente el impacto del tamaño del problema sobre la eficiencia de OpenMP en esquemas explícitos simples como FTCS. En particular, resulta relevante identificar umbrales prácticos a partir de los cuales la paralelización comienza a ser efectiva, así como evaluar la relación entre speedup, eficiencia y overhead bajo diferentes configuraciones de ejecución.

El objetivo de este trabajo es analizar cuantitativamente la eficiencia de la paralelización OpenMP aplicada a la ecuación de difusión unidimensional resuelta mediante el esquema FTCS. Para ello, se implementó un diseño experimental considerando diferentes tamaños de malla y configuraciones de hilos, evaluando métricas de desempeño como tiempo de ejecución, speedup, eficiencia paralela y overhead relativo. Adicionalmente, se verificó la conservación de la precisión numérica mediante normas de error y análisis de convergencia. Los resultados obtenidos permiten identificar umbrales prácticos de paralelización efectiva y establecer recomendaciones para implementaciones paralelas en problemas de difusión.

2. Trabajos Relacionados

2.1 Métodos numéricos para la ecuación de difusión

La resolución numérica de ecuaciones de difusión mediante diferencias finitas ha sido ampliamente estudiada debido a su importancia en física, ingeniería y ciencias computacionales. Los trabajos clásicos de Courant, Friedrichs y Lewy establecieron las bases de estabilidad para esquemas numéricos aplicados a ecuaciones diferenciales parciales [3]. Posteriormente, Von Neumann desarrolló herramientas sistemáticas para el análisis de estabilidad de esquemas explícitos e implícitos, consolidando fundamentos que continúan siendo utilizados en la actualidad [4].

Entre los métodos explícitos, el esquema FTCS destaca por su simplicidad matemática y facilidad de implementación computacional. Diversos estudios han evaluado sus propiedades de convergencia y estabilidad en problemas de difusión y advección-difusión [1], [2].

Aunque existen esquemas más avanzados y precisos, FTCS continúa siendo utilizado como referencia en estudios metodológicos y de desempeño computacional debido a su estructura sencilla y altamente paralelizable.

2.2 OpenMP y computación paralela

El desarrollo de arquitecturas multinúcleo ha impulsado el uso de modelos de programación paralela en aplicaciones científicas. OpenMP se ha consolidado como uno de los estándares más utilizados para programación en memoria compartida debido a su portabilidad y facilidad de integración en códigos existentes [5]. Su modelo basado en directivas permite paralelizar regiones específicas del código con modificaciones relativamente pequeñas en programas secuenciales.

Trabajos recientes han demostrado la utilidad de OpenMP en aplicaciones científicas de gran escala, particularmente en simulaciones numéricas y problemas multiphysics [6]. Sin embargo, la eficiencia de paralelización depende fuertemente del tamaño del

problema, la arquitectura de hardware y el overhead asociado a la sincronización y administración de hilos.

2.3 Métricas de desempeño paralelo

La evaluación del desempeño paralelo suele realizarse mediante métricas como speedup, eficiencia y overhead relativo. La Ley de Amdahl establece límites teóricos para el speedup alcanzable en función de la fracción serial del código [7]. Aunque este modelo proporciona una referencia útil, en aplicaciones reales intervienen además factores asociados a jerarquía de memoria, sincronización y comunicación entre hilos.

A pesar de la abundante literatura sobre programación paralela, todavía son limitados los estudios experimentales que analizan sistemáticamente la relación entre tamaño del problema y overhead en esquemas explícitos simples como FTCS. En este contexto, el presente trabajo busca proporcionar evidencia cuantitativa que permita identificar configuraciones prácticas de paralelización efectiva para problemas de difusión unidimensional.

3. Metodología

3.1 Modelo físico y ecuación de difusión

El problema estudiado corresponde a la ecuación de difusión unidimensional, utilizada para modelar procesos de transferencia de calor y transporte difusivo. La ecuación diferencial parcial considerada es: $\partial u/\partial t = D\,\partial^2 u/\partial x^2$. Donde $u(x,t)$ representa la variable física de interés y D es el coeficiente de difusión. El dominio espacial se definió en el intervalo $0 \le x \le L$.

La discretización numérica se realizó mediante el esquema FTCS (Forward Time Centered Space), utilizando diferencias finitas centradas para la derivada espacial y diferencias progresivas para la derivada temporal. La ecuación discreta utilizada fue:

$$u_i{}^{n+1} = u_i{}^n + r(u_{i+1}^n - 2u_i^n + u_{i-1}^n)$$

donde: $r = D\,\Delta t/\Delta x^2$ corresponde al parámetro de estabilidad del método.

Para garantizar estabilidad numérica se seleccionó un valor de $r = 0.45$, cumpliendo la condición de estabilidad del esquema FTCS. Las simulaciones se realizaron utilizando diferentes tamaños de malla espacial con el objetivo de analizar simultáneamente el comportamiento numérico y el desempeño computacional de la implementación paralela.

3.2 Implementación computacional

Se desarrollaron dos versiones del algoritmo numérico: una implementación serial de referencia y una implementación paralela utilizando OpenMP. Ambas versiones fueron programadas en lenguaje C, empleando la misma estructura numérica y las mismas condiciones iniciales y de frontera, con el propósito de garantizar comparaciones consistentes entre los resultados computacionales.

La paralelización se aplicó principalmente sobre el bucle espacial asociado al cálculo de la evolución temporal del esquema FTCS, debido a que cada punto de la malla depende únicamente de sus vecinos inmediatos. Para ello se utilizaron directivas #pragma omp parallel for con planificación estática (schedule(static)), apropiada para cargas de trabajo uniformes.

Adicionalmente, se paralelizaron regiones relacionadas con la inicialización de arreglos y el cálculo de métricas de error numérico. En el caso de las normas de error, se emplearon operaciones de reducción (reduction) para evitar condiciones de carrera y garantizar la consistencia de los resultados obtenidos durante la ejecución paralela.

La implementación fue compilada utilizando GCC con soporte OpenMP y optimizaciones de compilación orientadas a maximizar el desempeño computacional. Asimismo, se mantuvieron constantes las condiciones de compilación y ejecución durante todos los experimentos para asegurar la reproducibilidad de las mediciones.

Con el propósito de favorecer la reproducibilidad de los resultados, el código fuente utilizado durante los experimentos se encuentra disponible en un repositorio público, incluyendo las implementaciones serial y paralela, así como los scripts empleados para el procesamiento y visualización de datos.

3.3 Configuración experimental

Con el objetivo de analizar sistemáticamente el impacto de la paralelización sobre el desempeño computacional, se implementó un diseño experimental factorial considerando diferentes tamaños de malla y configuraciones de hilos OpenMP. La Tabla 1 resume los parámetros principales utilizados durante los experimentos. Los tamaños de malla evaluados fueron:

$Nx = 200,\ 1000,\ 5000,\ 10000,\ 20000$ mientras que las configuraciones paralelas consideradas correspondieron a: $p = 1,\ 2,\ 4,\ 8$ hilos de ejecución.

Tabla 1. Configuraciones experimentales utilizadas durante las simulaciones numéricas.

Parámetro	Valores
Nx	200 – 20000
Hilos	1, 2, 4, 8
(D)	0.1
(L)	1

IJISTA, Vol. II, Núm. I, junio 2026, ISSN 3122-4474

Todas las simulaciones se realizaron manteniendo constantes los parámetros físicos del problema. El dominio espacial se definió con longitud $L = 1.0$, coeficiente de difusión $D = 0.1$ y tiempo final de simulación $T_f = 0.2$. El parámetro de estabilidad se fijó en $r = 0.45$ para todas las configuraciones experimentales.

Los experimentos se ejecutaron sobre un procesador Intel Core i7-10700K con arquitectura multinúcleo y memoria compartida. Para reducir variaciones asociadas al sistema operativo y al escalamiento dinámico de frecuencia, se mantuvieron constantes las condiciones de ejecución durante todas las pruebas experimentales.

Con el fin de minimizar fluctuaciones estadísticas, cada configuración experimental fue ejecutada múltiples veces y se utilizaron los valores promedio para el análisis de desempeño. Esta estrategia permitió obtener mediciones más estables y comparables entre las diferentes configuraciones evaluadas.

3.4 Instrumentación y métricas de desempeño

Con el propósito de garantizar mediciones reproducibles y comparables, se implementó un esquema de instrumentación orientado tanto al análisis del rendimiento computacional como a la evaluación de la precisión numérica de las simulaciones. Las métricas seleccionadas permiten cuantificar el impacto de la paralelización OpenMP sobre el tiempo de ejecución, el overhead asociado y la conservación de la exactitud numérica respecto a la solución analítica.

Para caracterizar el desempeño computacional se utilizaron métricas de tiempo de pared y tiempo de CPU. El tiempo de pared fue medido mediante la función omp_get_wtime(), la cual proporciona el tiempo real transcurrido durante la ejecución del programa, incluyendo costos de sincronización y administración de hilos. Por otro lado, el tiempo de CPU se obtuvo mediante la función clock(), utilizada para estimar el tiempo efectivo de procesamiento consumido por el sistema.

La evaluación de precisión numérica se realizó utilizando las normas de error L_2 y L_∞, ampliamente empleadas en análisis numérico para comparar soluciones aproximadas con soluciones analíticas. El error global promedio se calculó mediante:

$$errL_2 = \sqrt{\sum_i (u_i - u_{exact}(x_i))^2 \Delta x}$$

Donde u_i representa la solución numérica en el nodo espacial i, mientras que $u_{exact}(x_i)$ corresponde a la solución analítica evaluada en la misma posición espacial.

El error máximo absoluto en toda la malla espacial se determinó mediante la norma:

$$errL_\infty = \max_i |u_i - u_{exact}(x_i)|$$

Estas métricas permiten caracterizar simultáneamente el desempeño computacional y la estabilidad numérica del algoritmo, proporcionando una base objetiva para comparar las implementaciones serial y paralela bajo diferentes configuraciones de tamaño de malla y número de hilos.

4. Resultados

4.1 Tiempo de ejecución y escalabilidad

La Figura 1 muestra la evolución del tiempo de ejecución en función del número de hilos para diferentes tamaños de malla. Debido a la amplia diferencia entre escalas temporales, se utilizó un eje vertical logarítmico con el propósito de visualizar simultáneamente configuraciones cuyos tiempos de ejecución difieren en varios órdenes de magnitud.

Para tamaños de malla pequeños, el incremento en el número de hilos provoca un aumento progresivo en el tiempo de ejecución. Este comportamiento indica que el overhead asociado a la creación, sincronización y administración de hilos domina completamente el costo computacional del algoritmo, haciendo que la versión paralela resulte menos eficiente que la implementación serial.

A medida que el tamaño de malla aumenta, el impacto relativo del overhead disminuye gradualmente. En particular, para $Nx = 20{,}000$ se observa una reducción parcial del tiempo de ejecución al utilizar 4 hilos, lo que evidencia el inicio de un régimen donde la paralelización comienza a ofrecer beneficios prácticos de rendimiento.

La Tabla 2 resume las configuraciones más representativas obtenidas durante los experimentos. Los resultados muestran que el máximo speedup alcanzado fue de 1.05× para $N_x = 20{,}000$ con 4 hilos, mientras que para problemas pequeños la paralelización produjo degradación significativa del desempeño computacional.

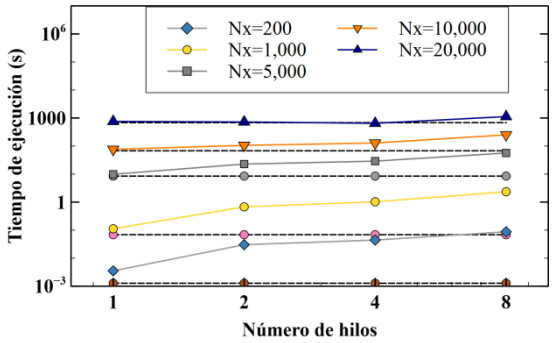


Fig. 1. Tiempo de ejecución en función del número de hilos para diferentes tamaños de malla. En los dos ejes se utiliza escala logarítmica.

Tabla 2. Resultados representativos de desempeño para diferentes tamaños de malla y configuraciones de ejecución.

N_x	Mejor configuración	Tiempo mínimo (s)	Speedup máximo	Eficiencia máxima
200	Serial	0.001275	0.36	36.0%
1,000	Serial	0.068978	0.62	62.0%
5,000	Serial	8.476183	0.86	86.1%
1,0000	Serial	67.801847	0.89	89.0%
2,0000	4 hilos	647.830333	1.05	26.3%

4.2 Speedup y eficiencia paralela

La Figura 2 presenta el speedup obtenido para las diferentes configuraciones de tamaño de malla y número de hilos. El comportamiento observado muestra una fuerte dependencia entre el desempeño paralelo y el tamaño del problema, evidenciando que la paralelización no produce mejoras uniformes en todas las configuraciones experimentales.

Para tamaños de malla pequeños, el speedup permanece por debajo de la unidad incluso al incrementar el número de hilos, indicando que la versión paralela resulta más lenta que la implementación serial. Este comportamiento se encuentra asociado al overhead introducido por OpenMP, el cual domina el costo computacional cuando la carga de trabajo por hilo es reducida.

A medida que el tamaño de la malla aumenta, el speedup mejora progresivamente. En particular, para $N_x = 20{,}000$ se alcanzó un speedup máximo de aproximadamente 1.05× utilizando 4 hilos, constituyendo la primera configuración donde la paralelización produjo una reducción efectiva del tiempo de ejecución.

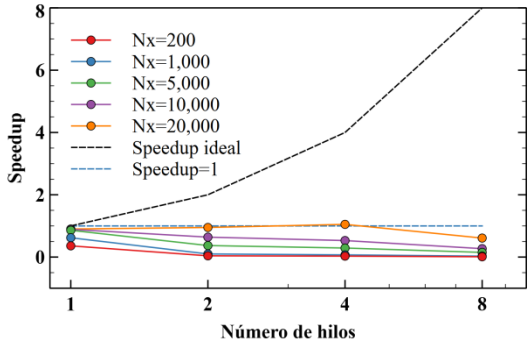


Fig. 2. Speedup en función del número de hilos para diferentes tamaños de malla. La línea discontinua representa el comportamiento ideal S=p.

La Figura 3 muestra la eficiencia paralela correspondiente a las mismas configuraciones experimentales. Se observa una disminución sistemática de la eficiencia conforme aumenta el número de hilos, comportamiento típico en aplicaciones paralelas donde los costos de sincronización y acceso compartido a memoria crecen con el paralelismo empleado.

Aunque la eficiencia disminuye para configuraciones con mayor número de hilos, los resultados muestran que el impacto relativo del overhead se reduce conforme el tamaño del problema incrementa. Esto indica que problemas con mayor carga computacional poseen un mejor potencial de aprovechamiento de arquitecturas multinúcleo mediante OpenMP.

4.3 Evolución del overhead

La Figura 4 muestra la evolución del overhead relativo en función del tamaño de malla para la configuración de 8 hilos. Se observa una disminución sistemática del overhead conforme aumenta el tamaño del problema, indicando que el costo relativo asociado a la paralelización se amortiza progresivamente para cargas computacionales más grandes.

Para tamaños de malla pequeños, el overhead domina completamente el tiempo de ejecución total, provocando que la implementación paralela resulte considerablemente más lenta que la versión serial. Este comportamiento se encuentra relacionado con los costos de creación y sincronización de hilos, así como con la administración de memoria compartida realizada por OpenMP.

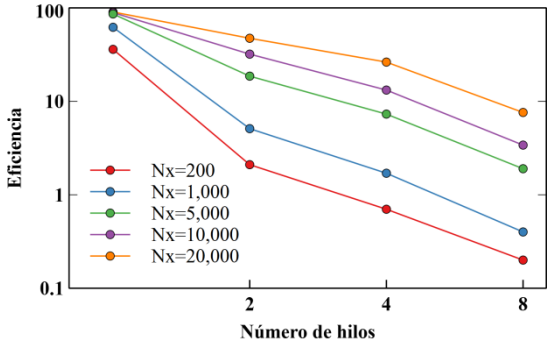


Fig. 3. Eficiencia paralela en función del número de hilos para diferentes tamaños de malla.

A medida que el número de puntos espaciales aumenta, el tiempo dedicado al cálculo numérico comienza a superar el costo administrativo de paralelización. Como consecuencia, el overhead relativo disminuye gradualmente, permitiendo que el algoritmo aproveche de manera más eficiente los recursos multinúcleo disponibles.

Aunque el overhead disminuye para problemas grandes, los resultados muestran que su impacto continúa siendo significativo incluso para las configuraciones de mayor tamaño evaluadas. Esto evidencia que, además de la fracción paralelizable del código, el rendimiento práctico depende fuertemente de factores asociados a

IJISTA, Vol. II, Núm. I, junio 2026, ISSN 3122-4474

sincronización, acceso a memoria y distribución de carga entre hilos.

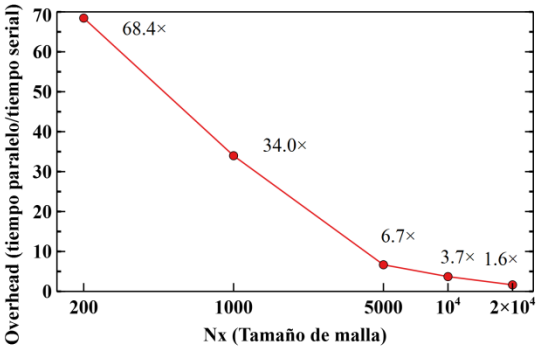


Fig. 4. Evolución del overhead relativo para la configuración de 8 hilos en función del tamaño de malla.

4.4 Precisión numérica y convergencia

Además del análisis de desempeño computacional, se evaluó la precisión numérica de la implementación mediante el estudio de convergencia del esquema FTCS. La Figura 5 muestra la evolución del error numérico conforme se refina la malla espacial, utilizando la norma L2 como métrica de comparación respecto a la solución analítica.

Los resultados muestran una disminución sistemática del error al incrementar la resolución espacial, indicando que la solución numérica converge correctamente hacia la solución exacta del problema de difusión. Este comportamiento confirma la estabilidad y consistencia de la implementación utilizada durante los experimentos.

La pendiente observada en la gráfica de convergencia es consistente con un comportamiento aproximado de segundo orden espacial, en concordancia con las propiedades teóricas del esquema FTCS utilizado para discretizar la ecuación de difusión.

Adicionalmente, no se observaron diferencias significativas entre las soluciones obtenidas mediante las implementaciones serial y paralela, presentándose únicamente variaciones del orden de precisión de máquina. Esto indica que la paralelización mediante OpenMP conserva adecuadamente la exactitud numérica del algoritmo original.

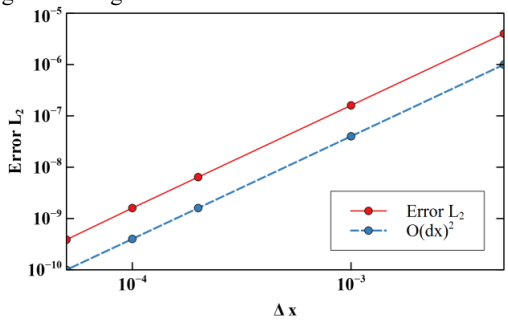


Fig. 5. Convergencia del error numérico en norma L_2. La línea punteada representa la tendencia teórica $O(\Delta x^2)$.

5. Análisis de Resultados

5.1 Interpretación del comportamiento paralelo

Los resultados obtenidos muestran que la eficiencia de paralelización depende fuertemente del tamaño del problema computacional. Para tamaños de malla pequeños, el costo asociado a la creación y sincronización de hilos domina el tiempo total de ejecución, provocando que la implementación paralela resulte menos eficiente que la versión serial.

El comportamiento observado es consistente con el hecho de que el esquema FTCS utilizado posee operaciones computacionales relativamente simples por iteración. En consecuencia, cuando el número de puntos espaciales es reducido, la cantidad de trabajo asignada a cada hilo no resulta suficiente para compensar el overhead introducido por OpenMP.

A medida que el tamaño de la malla aumenta, el costo computacional asociado al cálculo numérico comienza a predominar sobre el overhead de paralelización. Esto explica la mejora gradual observada en el speedup y la reducción relativa del overhead para configuraciones con mayor número de puntos espaciales.

En particular, la configuración correspondiente a $N_x = 20{,}000$ y 4 hilos representó el primer caso donde la paralelización produjo una mejora efectiva de rendimiento respecto a la versión serial. Este comportamiento sugiere la existencia de un umbral práctico de tamaño de problema a partir del cual OpenMP comienza a utilizar de manera más eficiente los recursos multinúcleo disponibles.

5.2 Relación con la Ley de Amdahl

El comportamiento experimental observado puede interpretarse mediante la Ley de Amdahl, la cual establece que el speedup máximo alcanzable se encuentra limitado por la fracción serial del código. Aunque el algoritmo FTCS posee una estructura altamente paralelizable debido a la independencia espacial de sus operaciones, el desempeño práctico continúa condicionado por costos asociados a sincronización y administración de hilos.

A partir de los resultados experimentales correspondientes a las configuraciones de mayor tamaño de malla, se estimó que la fracción paralelizable del código es considerablemente alta. Sin embargo, el speedup obtenido experimentalmente permanece muy por debajo del comportamiento ideal, evidenciando que el overhead práctico introduce limitaciones importantes que no son capturadas completamente por el modelo teórico idealizado.

La diferencia observada entre el speedup ideal y el speedup experimental confirma que el rendimiento paralelo no depende únicamente de la proporción de código paralelizable. Factores asociados a jerarquía de memoria, acceso concurrente a recursos compartidos y sincronización entre hilos desempeñan un papel fundamental en el desempeño final de aplicaciones científicas paralelas.

Los resultados obtenidos muestran que incluso algoritmos con una estructura computacional favorable para paralelización pueden presentar ganancias limitadas cuando el tamaño del problema no es suficientemente grande. Esto resalta la importancia de evaluar experimentalmente el comportamiento de aplicaciones paralelas bajo condiciones reales de ejecución.

5.3 Implicaciones prácticas

Los resultados obtenidos permiten establecer recomendaciones prácticas para el uso de OpenMP en problemas de difusión resueltos mediante FTCS. Para tamaños de malla pequeños, la implementación serial resulta más conveniente debido a que el overhead de paralelización supera ampliamente cualquier posible ganancia de rendimiento.

En configuraciones con mayor carga computacional, la paralelización comienza a mostrar beneficios parciales, particularmente cuando se utilizan cantidades moderadas de hilos. En este contexto, configuraciones intermedias como 4 hilos mostraron el mejor equilibrio entre costo de sincronización y distribución de carga computacional.

Estos resultados sugieren que la eficiencia de OpenMP depende no solamente del número de hilos disponibles, sino también de la relación entre el tamaño del problema y el costo administrativo asociado a la paralelización. En consecuencia, la selección de configuraciones paralelas adecuadas debe realizarse considerando simultáneamente el hardware disponible y la carga computacional del problema estudiado.

5.4 Limitaciones del estudio

El presente trabajo se enfocó en una implementación OpenMP ejecutada sobre una única arquitectura de hardware basada en memoria compartida, por lo que los resultados obtenidos podrían variar en sistemas con diferentes configuraciones de procesador, jerarquía de caché o ancho de banda de memoria.

Asimismo, el análisis se limitó a la ecuación de difusión unidimensional utilizando el esquema explícito FTCS, el cual posee una estructura computacional relativamente simple y altamente paralelizable. En problemas multidimensionales, esquemas implícitos o arquitecturas heterogéneas, el comportamiento del overhead y la eficiencia paralela podrían diferir significativamente.

A pesar de estas limitaciones, los resultados presentados proporcionan evidencia cuantitativa útil para comprender el impacto práctico de la paralelización OpenMP en problemas numéricos de difusión y constituyen una base experimental para estudios futuros en arquitecturas y esquemas numéricos más complejos.

6. Conclusiones

En este trabajo se realizó un análisis cuantitativo de la eficiencia de paralelización OpenMP aplicada a la ecuación de difusión unidimensional resuelta mediante el esquema FTCS. Los resultados obtenidos mostraron que el desempeño paralelo depende fuertemente del tamaño del problema computacional y del número de hilos utilizados durante la ejecución.

Para tamaños de malla pequeños, la paralelización resultó contraproducente debido al impacto dominante del overhead asociado a sincronización y administración de hilos. En contraste, para configuraciones con mayor carga computacional se observó una reducción parcial del tiempo de ejecución, alcanzándose un speedup máximo aproximado de 1.05× para $N_x = 20{,}000$ utilizando 4 hilos.

El análisis de convergencia confirmó que la implementación conserva adecuadamente la precisión numérica del esquema FTCS, presentando comportamiento consistente con un orden espacial aproximado de segundo orden. Asimismo, no se observaron diferencias significativas entre las soluciones obtenidas mediante las implementaciones serial y paralela.

Los resultados obtenidos evidencian que la eficiencia práctica de OpenMP no depende únicamente de la fracción paralelizable del código, sino también del balance entre carga computacional y overhead de paralelización. En este sentido, el trabajo proporciona evidencia experimental útil para la selección de configuraciones paralelas apropiadas en aplicaciones numéricas basadas en diferencias finitas.

Como trabajo futuro, se propone extender el análisis a problemas bidimensionales y tridimensionales, así como evaluar esquemas implícitos y arquitecturas heterogéneas que permitan estudiar con mayor profundidad el comportamiento del overhead y la escalabilidad en aplicaciones científicas paralelas.

Los autores agradecen el apoyo otorgado a esta investigación por parte del Centro Nacional de Supercómputo del IPICYT, A.C., mediante el tiempo de cómputo otorgado al proyecto TKII-E-1025-I-081025-75 y TKII-E-1025-I-081025-76.

7. Referencias

[1] S. Sungnul, K. Para, E. J. Moore, S. Sirisubtawee y S. Phongthanapanich, «A finite difference method for solution of integer-order and Caputo fractional-time advection-diffusion-reaction equations: Convergence analysis and application to air pollution,» *Engineering Letters,* vol. 33, nº 2, p. 292–311, 2025.

[2] C. L. Gardner, «Numerical Methods for Parabolic PDEs,» de *Numerical Methods for Unsteady Compressible Flow Problems*, Springer, 2024.

[3] R. Courant, K. Friedrichs y H. Lewi, «Über die partiellen Differenzengleichungen der mathematischen Physik,» *Mathematische Annalen,* vol. 100, nº 1, p. 32–74, 1928.

[4] J. von Neumann y R. D. Richtmyer, «A method for the numerical calculation of hydrodynamic shocks,» *Journal of Applied Physics,* vol. 21, nº 3, p. 232–237, 1950.

[5] Intel Corporation, «"OpenMP and the Future of Scientific Computing," Intel Developer Blog,» 2025. [En línea]. Available: https://www.intel.com.

[6] M. Thavappiragasam, J. A. Harris, E. Endeve y B. Videau, «Performance porting the ExaStar multi-physics app Thornado on heterogeneous systems: A Fortran-OpenMP code-base evaluation," in Advancing OpenMP for Future Accelerators,» de *20th International Workshop on OpenMP.*, Cham, Switzerland, 2024.

[7] G. M. Amdahl, «Validity of the single processor approach to achieving large scale computing capabilities,» de *AFIPS Spring Joint Computer Conference*, 1967.

[8] J. L. Gustafson, «Reevaluating Amdahl's law,» *Communications of the ACM,* vol. 31, nº 5, p. 532–533, 1988.

[9] Á. Nagy, «New analytical results and comparison of 14 numerical schemes for the diffusion equation with space-dependent diffusion coefficient,» *Mathematics,* 2024.

[10] Z. Xiang y T. G. Robertazzi, «A DLT-aware performance evaluation framework for virtual-core speedup modeling,» *Future Internet,* vol. 17, nº 11, 2025.

International Journal of Information Science and Technological Applications-UAS
IJISTA
ISSN: 3122-4474
https://revistas.uas.edu.mx/index.php/IJISTA

Junio 2026
Vol. II.
Número. II

Implementación de Ciberseguridad en Redes mediante MikroTik y Servicios en Debian

Implementation of Cybersecurity in Networks using MikroTik and Services in Debian

Denisse Carolina Navarro Garcia[1], Sayda Karely Carmona Medina[1], Henry Osuna Boltor[1]

[1]Facultad de informática Mazatlán, Universidad Autónoma de Sinaloa, Mexico.

https://orcid.org/0009-0003-9639-8227
pro593737@gmail.com

https://orcid.org/0009-0004-7842-2948
hosuna2345@gmail.com

Autor de Correspondencia: Denisse Carolina Navarro García, denissega908@gmail.com,
https://orcid.org/0009-0007-2489-0414

Recibido: abril 2026
Publicado: junio 2026

IJISTA, Vol. II, Núm. II, Junio 2026,
ISSN: 3122-4474

Resumen:

Este trabajo presenta la implementación de ciberseguridad en redes utilizando MikroTik y un servidor Debian, con el objetivo de mejorar la protección ante amenazas como accesos no autorizados y ataques informáticos. En la introducción se destaca la importancia de la seguridad en redes modernas y el uso de herramientas accesibles para su protección. En los trabajos relacionados se mencionan enfoques como la seguridad, el uso de firewalls, el monitoreo de red y la comunicación segura. La metodología consistió en crear un entorno con IP pública, configurar un servidor Debian con distintos servicios y aplicar reglas de firewall en MikroTik, además de implementar HTTPS con Certbot. Los resultados muestran que inicialmente el sistema era vulnerable, pero después de aplicar las configuraciones se logró restringir accesos y mejorar la seguridad. El análisis confirma que el uso de firewall, cifrado y monitoreo permite un mejor control del entorno. Finalmente, se concluye que la combinación de estas herramientas es fundamental para lograr una red segura y funcional.

Palabras Clave:

Ciberseguridad, MikroTik, Debian, Firewall, Redes, Seguridad perimetral, HTTPS, Certbot, Monitoreo de red, VirtualHosts.

Abstract:

This paper presents the implementation of cybersecurity measures in networks using MikroTik and a Debian server, with the aim of improving protection against threats such as unauthorized access and cyberattacks. The introduction highlights the importance of security in modern networks and the use of accessible tools for their protection. Related works mention approaches such as perimeter security, the use of firewalls, network monitoring, and secure communication. The methodology consisted of creating an environment with a public IP address, configuring a Debian server with various services, and applying firewall rules in MikroTik, in addition to implementing HTTPS with Certbot. The results show that the system was initially vulnerable, but after applying the configurations, access was restricted and security was improved. The analysis confirms that the use of firewalls, encryption, and monitoring allows for better control of the environment. Finally, it is concluded that the combination of these tools is fundamental to achieving a secure and functional network.

Keywords:

Cybersecurity, MikroTik, Debian, Firewall, Networks, Perimeter Security, HTTPS, Certbot, Network Monitoring, VirtualHosts.

1. Introducción

La ciberseguridad representa uno de los principales desafíos en la administración de redes modernas, especialmente debido al crecimiento constante de servicios expuestos a Internet y al incremento de amenazas informáticas como accesos no autorizados, escaneo de puertos, robo de información y ataques dirigidos a infraestructuras conectadas.

En entornos académicos y organizacionales, la implementación de mecanismos de protección resulta fundamental para garantizar la confidencialidad, integridad y disponibilidad de los servicios de red. En este contexto, tecnologías como MikroTik y sistemas basados en Linux han adquirido relevancia debido a su flexibilidad, bajo costo y capacidad para integrar funciones de administración, monitoreo y seguridad dentro de una misma infraestructura.

MikroTik, mediante RouterOS, permite implementar mecanismos de seguridad perimetral, reglas de firewall, control de acceso y filtrado de tráfico, mientras que servidores basados en Debian ofrecen una plataforma estable para el despliegue de servicios y herramientas de monitoreo de red. La combinación de estas tecnologías permite construir infraestructuras funcionales orientadas a mejorar la protección de servicios expuestos a Internet.

El presente trabajo tiene como objetivo implementar y analizar un entorno de red seguro utilizando un servidor Debian y un dispositivo MikroTik, integrando mecanismos de control de acceso, comunicación cifrada y monitoreo de servicios. Para ello, se configuraron herramientas como FOG, ntopng y Zabbix, además de reglas de firewall y certificados HTTPS mediante Certbot.

A diferencia de implementaciones enfocadas únicamente en mecanismos individuales de seguridad, el presente trabajo integra múltiples herramientas de protección, monitoreo y administración dentro de una misma infraestructura basada en MikroTik y Debian. Esta integración permite evaluar el comportamiento conjunto de distintas capas de seguridad en un entorno funcional y de bajo costo orientado a escenarios académicos y organizacionales.

Asimismo, se realizaron pruebas de conectividad, validación de accesibilidad y análisis de puertos para evaluar el comportamiento del sistema antes y después de la implementación de los mecanismos de seguridad. De esta manera, el estudio busca analizar el impacto de la integración de mecanismos de seguridad perimetral, cifrado y monitoreo sobre la reducción de exposición de servicios y el control de accesos en infraestructuras de red con administración centralizada.

2. Trabajos Relacionados

El análisis de trabajos relacionados permite contextualizar la implementación de mecanismos de ciberseguridad en redes, destacando las principales tecnologías, enfoques y soluciones utilizadas en entornos reales. En particular, el uso de dispositivos de red, herramientas de monitoreo y protocolos de comunicación segura ha permitido mejorar la protección de infraestructuras expuestas a internet.

A través de la revisión de la literatura, se identifican tendencias clave como la implementación de seguridad perimetral mediante firewalls, el uso de sistemas de detección de intrusos, la aplicación de cifrado en la comunicación y el monitoreo continuo de la red. En las siguientes subsecciones se presentan los principales avances en estas áreas.

2.1 Seguridad perimetral en redes

La seguridad perimetral constituye uno de los elementos fundamentales en la protección de redes, permitiendo controlar el tráfico entrante y saliente mediante políticas de filtrado. Diversos estudios han documentado el uso de listas de control de acceso (ACL), segmentación de red y redes privadas virtuales para restringir accesos no autorizados. En este contexto, la optimización de reglas de filtrado de paquetes se ha consolidado como una estrategia crítica para fortalecer el perímetro en infraestructuras de pequeña y mediana escala [1].

Asimismo, los firewalls avanzados permiten inspeccionar paquetes y prevenir intrusiones, representando una barrera crítica en entornos con servicios expuestos a internet facilitando la identificación temprana de anomalías en el tráfico [2].

2.2 Plataformas MikroTik para control de red

Las soluciones basadas en MikroTik han ganado relevancia debido a su flexibilidad y bajo costo, permitiendo implementar funciones como firewall, NAT y control de tráfico en redes de distintos tamaños. Diversas investigaciones han demostrado que el análisis de rendimiento de las reglas de firewall en RouterOS es fundamental para garantizar una gestión eficiente del tráfico sin comprometer el rendimiento del hardware [3].

Asimismo, MikroTik ha sido utilizado en distintos entornos académicos y organizacionales para fortalecer la seguridad perimetral mediante filtrado de paquetes, control de acceso y administración centralizada del tráfico de red [4].

2.3 Detección de intrusos y análisis de tráfico

Los sistemas de detección de intrusos (IDS) y el análisis de anomalías en el tráfico de red han sido ampliamente estudiados como mecanismos complementarios a la seguridad perimetral. Estos sistemas permiten identificar comportamientos sospechosos y prevenir incidentes de seguridad [5,6].

Asimismo, investigaciones recientes han abordado técnicas avanzadas basadas en aprendizaje automático para mejorar la detección de amenazas en tiempo real [7,8].

2.4 Comunicación segura en redes

La protección de la información en tránsito es un aspecto fundamental en la ciberseguridad. En este sentido, el uso de protocolos seguros como HTTPS permite garantizar la confidencialidad e integridad de los datos transmitidos, mitigando ataques de interceptación como los de tipo “man-in-the-middle” [9].

2.5 Monitoreo y gestión de redes

El monitoreo continuo de la red permite supervisar el estado de los sistemas y detectar anomalías en tiempo real. Diversos estudios destacan la importancia de herramientas de monitoreo para la gestión eficiente de redes y la prevención de incidentes [10,11].

2.6 Tendencias en arquitecturas de red seguras

Las arquitecturas modernas de red, como las basadas en software-defined networking (SDN), han introducido nuevos enfoques para la gestión centralizada y flexible de la seguridad en redes [12,13].

A diferencia de los trabajos previamente mencionados, el presente proyecto integra múltiples mecanismos de seguridad en una misma infraestructura, combinando seguridad perimetral mediante MikroTik, servicios desplegados en un servidor Debian y herramientas de monitoreo dentro de un entorno con acceso mediante IP pública. Esto permitió evaluar el comportamiento conjunto de dichas tecnologías en un escenario de implementación funcional orientado a la protección y administración de servicios de red.

3. Metodología

En esta sección se describe el procedimiento para la implementación del entorno de red segura, incluyendo la configuración del servido, el uso de dispositivos de red y la aplicación de mecanismos de ciberseguridad. Asimismo, se integraron herramientas de monitoreo y gestión que permitieron validar el funcionamiento del sistema en un entorno más cercano a condiciones reales.

3.1. Entornos de Red y Configuración del Servidor

Para el desarrollo del proyecto se implementó un entorno de red con acceso externo mediante el uso de una IP pública. Se configuro un servidor basado en Debian Linux, el cual fue utilizado para alojar un servicio web y herramientas de monitoreo.

El servidor permitió simular una infraestructura de red donde múltiples servicios se encuentran disponibles y accesibles de forma controlada a través de internet. Esta configuración facilitó la evaluación de riesgos asociados a la exposición de servicios.

Adicionalmente, se empleó un dispositivo Mikrotik como elemento de control perimetral, encargado de gestionar el tráfico mediante reglas de firewall.

3.2. Implementación de Servicios en el Servidor

Sobre el servidor Debian se implementaron múltiples servicios orientados a la administración, monitoreo y gestión de la red:

- FOG Project: utilizado para la gestión y clonación de equipos en la red.
- Ntopng: herramienta empleada para el monitoreo y análisis del tráfico de red en tiempo real.
- Comandos ping: utilizado para verificar la conectividad entre dispositivos y validar la accesibilidad del servidor.

Para la organización de estos servicios, se configuró VirtualHosts en el servidor web, permitiendo el acceso a cada herramienta mediante nombre de dominio locales personalizados.

Por ejemplo, cada servicio fue asociado a un subdominio especifico, de manera que, al acceder desde la interfaz principal, el usuario era redirigido automáticamente a direcciones como:

- fog.sredes02
- ntopng.sredes02
- zabbix.sredes02

Esta configuración permitió una administración más clara y estructurada de los servicios implementados.

3.3. Herramientas y Configuración del Sistema

Para la implementación del entorno se utilizaron las siguientes herramientas:

- Debian Linux: sistema operativo base del servidor.
- Certbot: empleado para la generación de certificados digitales y la habilitación del protocolo HTTPS, garantizando la comunicación segura.
- Nmap (Network Mapper): utilizado para la identificación de puertos abiertos y servicios activos en el servidor.
- Herramientas de conectividad: se utilizaron comandos ping para verificar la comunicación entre dispositivos y validar la accesibilidad del servidor.
- Mikrotik Router: dispositivo utilizando para la administración de la red y la implementación de seguridad perimetral.

- Firewall en Mikrotik: configurado mediante reglas de filtrado de tráfico, permitiendo controlar el acceso a los servidores expuestos y bloquear conexiones no autorizadas.

Estas herramientas y configuraciones permitieron construir un entorno funcional, en el cual se integraron tanto servicios de red como mecanismos de protección, asegurando la disponibilidad y seguridad del sistema.

3.4. Configuración de Seguridad en Mikrotik

El dispositivo MikroTik fue configurado como el elemento principal de seguridad perimetral, encargado de gestionar y filtrar el tráfico de red entre la red interna y el acceso mediante IP pública.

Para ello, se implementaron reglas de firewall orientadas al control de acceso y protección del servidor Debian expuesto, siguiendo las recomendaciones y lineamientos establecidos en la documentación oficial de RouterOS y Firewall Filter de MikroTik [14,15]. Estas reglas permitieron establecer políticas específicas basadas en direcciones IP, puertos y protocolos.

En primer lugar, se definieron reglas para permitir únicamente el tráfico necesario hacia los servicios publicados, principalmente en los puertos correspondientes a HTTP y HTTPS. Esto aseguró que el acceso externo se limitara únicamente a los servicios web habilitados en el servidor.

Posteriormente, se configuraron reglas para bloquear conexiones no autorizadas provenientes de direcciones IP externas, evitando accesos indebidos a servicios no expuestos. Asimismo, se restringió el acceso a puertos no utilizados, reduciendo la superficie de ataque del sistema.

Adicionalmente, se implementaron reglas orientadas a mitigar intentos de escaneo de puertos, bloqueando tráfico sospechoso y conexiones repetitivas que pudieran indicar actividades maliciosas.

Estas configuraciones permitieron establecer un control efectivo del tráfico entrante y saliente, garantizando que únicamente las solicitudes legítimas fueran procesadas por el servidor.

Como resultado, el MikroTik funcionó como una barrera de seguridad que protege el entorno, asegurando que los servicios implementados permanezcan accesibles únicamente bajo condiciones controladas.

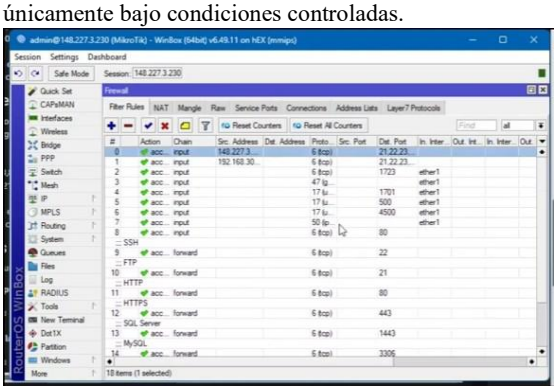


Figura 1. Configuración de reglas de firewall en MikroTik para el control de acceso al servidor. Elaboración propia.

3.5. Implementación de Comunicación Segura

Se implementó un mecanismo de cifrado mediante el uso de Certbot, habilitando el protocolo HTTPS en el servidor.

Esto permitió asegurar la comunicación entre el cliente y los servicios desplegados, garantizando la confidencialidad de los datos y protegiendo contra posibles ataques de interceptación.

3.6. Procesamiento de Pruebas

El proceso de validación se llevó a cabo en las siguientes fases:

Verificación de Conectividad: Se utilizo el comando ping para comprobar la comunicación entre dispositivos y validar la accesibilidad mediante la IP pública.

Evaluación de Servicios: Se verificó el correcto funcionamiento de los servicios implementados (FOG, ntopng y Zabbix), así como la correcta redirección mediante VirtualHosts.

Aplicación y Validación de Seguridad: Se implementaron reglas de firewall en MikroTik y posteriormente se realizaron pruebas de acceso para confirmar la restricción de tráfico no autorizado.

Durante este proceso, se analizaron distintos escenarios de implementación práctica, lo que permitió validar la efectividad de las configuraciones dentro de la arquitectura desplegada.

3.7. Métricas de Evaluación y Validación

Con el objetivo de validar la efectividad del modelo de seguridad implementado, se establecieron distintos criterios de evaluación orientados a medir el comportamiento de la infraestructura antes y después de la configuración de los mecanismos de protección.

Las pruebas realizadas se enfocaron en la accesibilidad de los servicios, la restricción de tráfico no autorizado y la disponibilidad segura de los sistemas desplegados.

3.7.1. Verificación de conectividad

Inicialmente se realizaron pruebas de conectividad mediante el comando ping para comprobar la comunicación entre dispositivos dentro de la red y validar el acceso a través de la dirección IP pública configurada.

Posteriormente, se evaluó el comportamiento de la red después de aplicar las reglas de firewall en MikroTik, verificando la restricción de accesos no autorizados y el filtrado del tráfico.

3.7.2. Evaluación de puertos y accesibilidad

Se analizaron los servicios accesibles antes y después de la implementación de las políticas de seguridad. Antes de aplicar las reglas de firewall, múltiples puertos del servidor podían responder a conexiones externas. Después de la configuración, únicamente se permitió el acceso a los servicios autorizados mediante HTTPS.

Además, los escaneos realizados mediante Nmap mostraron una reducción en la cantidad de puertos accesibles desde la red externa, permitiendo únicamente el acceso a los servicios autorizados y disminuyendo la exposición del servidor frente a posibles accesos no autorizados.

3.7.3. Validación de comunicación segura

Se verificó el correcto funcionamiento del protocolo HTTPS mediante la implementación de certificados digitales utilizando Certbot. Esto permitió validar que la comunicación entre el cliente y el servidor se realizara de forma cifrada y segura.

3.7.4. Monitoreo y disponibilidad de servicios

Finalmente, se comprobó la disponibilidad y funcionamiento de los servicios implementados, incluyendo FOG, ntopng y Zabbix, verificando el acceso mediante VirtualHosts configurados en el servidor Debian.

Tabla 1. Comparación antes y después de la implementación.
Elaboración propia.

Parámetro evaluado	**Antes de la implementación**	**Después de la implementación**
Comunicación segura	HTTP	HTTPS
Acceso a servicios	Sin restricciones	Restringido mediante firewall
Puertos accesibles	Múltiples puertos abiertos	Solo puertos autorizados
Monitoreo de red	No implementado	Implementado con Zabbix y ntopng
Protección perimetral	No configurada	Reglas de firewall activas

4. Resultados

Los resultados obtenidos reflejan la implementación exitosa de un entorno de red funcional y seguro, en el cual se integraron múltiples servicios y mecanismos de protección bajo condiciones similares a un escenario real.

Inicialmente, el servidor Debian configurado con una dirección IP pública presentaba accesibilidad directa desde la red externa, lo que evidenciaba una exposición potencial a accesos no autorizados. Sin embargo, tras la implementación de reglas de firewall en el dispositivo MikroTik, se logró restringir el acceso únicamente a los servicios permitidos, reduciendo significativamente la superficie de ataque.

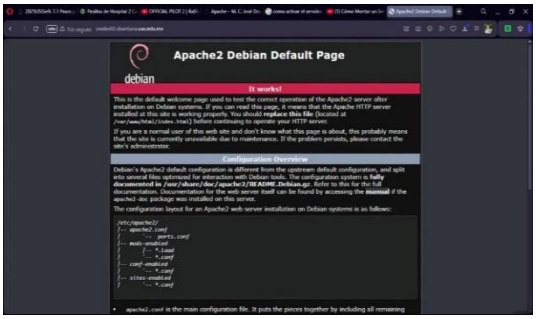


Figura 2. Acceso inicial al servidor Debian antes de la implementación de certificados HTTPS. Elaboración propia.

La aplicación de políticas de filtrado permitió bloquear conexiones no autorizadas y limitar el tráfico a puertos específicos, principalmente aquellos destinados a servicios web. Esto se validó mediante pruebas de conectividad, donde únicamente las solicitudes permitidas lograron establecer comunicación con el servidor.

Adicionalmente, los escaneos realizados mediante Nmap mostraron una reducción en la cantidad de puertos accesibles desde la red externa después de aplicar las reglas de firewall, permitiendo únicamente el acceso a los servicios autorizados.

Asimismo, la implementación de comunicación segura mediante el uso de Certbot permitió habilitar el protocolo HTTPS, garantizando la confidencialidad de los datos transmitidos. Esto aseguró que la interacción entre el cliente y el servidor se realizara de forma cifrada, reduciendo riesgos de interceptación.

En cuanto a los servicios desplegados, se verificó el correcto funcionamiento de herramientas como FOG, ntopng y Zabbix, las cuales fueron accesibles mediante nombres de dominio configurados a través de VirtualHosts. Esta organización permitió una navegación estructurada, donde cada servicio se encuentra claramente identificado y separado dentro del mismo servidor.

La interfaz web implementada funcionó como punto central de acceso, permitiendo redirigir a cada uno de los servicios de forma eficiente. Esto facilitó la administración del sistema y evidenció la correcta integración entre los componentes.

IJISTA, Vol. II, Núm. II, junio 2026, ISSN 3122-4474

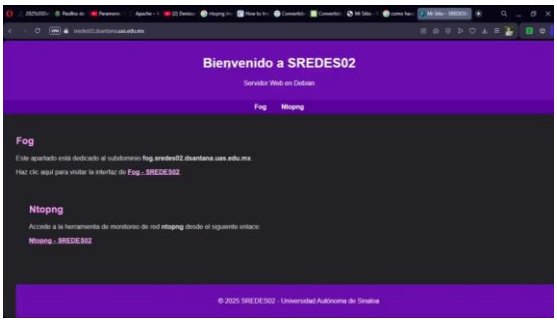


Figura 3. Interfaz web del sistema implementado, mostrando el acceso a los servicios FOG, ntopng y Zabbix mediante VirtualHosts. Elaboración propia.

En conjunto, los resultados demuestran que la combinación de un servidor Debian, configuraciones de seguridad en MikroTik y mecanismos de cifrado permiten construir una infraestructura de red segura, funcional y alineada con prácticas actuales de ciberseguridad, manteniendo un equilibrio adecuado entre accesibilidad y protección.

5. Análisis de Resultados

El análisis de los resultados obtenidos permite identificar el impacto real de las configuraciones implementadas en la seguridad y funcionamiento del entorno de red.

En primer lugar, la exposición inicial del servidor Debian mediante una dirección IP pública evidenció un escenario vulnerable, en el cual múltiples servicios podían ser accesibles sin restricciones. Esta condición refleja una problemática común en entornos mal configurados, donde la falta de controles de seguridad incrementa significativamente el riesgo de ataques.

La implementación de reglas de firewall en MikroTik permitió mejorar el control del tráfico. A partir de estas configuraciones, se observó una reducción en la accesibilidad de servicios no autorizados, lo que indica un control más estricto del tráfico de red. Este resultado confirma que la seguridad perimetral es un elemento fundamental para la protección de infraestructuras expuestas a internet.

Por otro lado, la habilitación del protocolo HTTPS mediante el uso de Certbot permitió asegurar la comunicación entre cliente y servidor. Este mecanismo no solo protege la información transmitida, sino que también incrementa la confianza en el sistema al garantizar la autenticidad del servidor. En este sentido, el cifrado se posiciona como un complemento esencial de las políticas de seguridad.

Asimismo, la integración de herramientas como FOG, ntopng y Zabbix aportó un valor adicional al sistema, al permitir no solo la prestación de servicios, sino también su monitoreo y administración. En particular, el uso de plataformas de monitoreo facilita la detección de anomalías y la supervisión continua del estado de la red, lo cual es clave para la prevención de incidentes.

Figura 4. Interfaz de acceso al servicio FOG Project implementado en el servidor Debian. Elaboración propia.

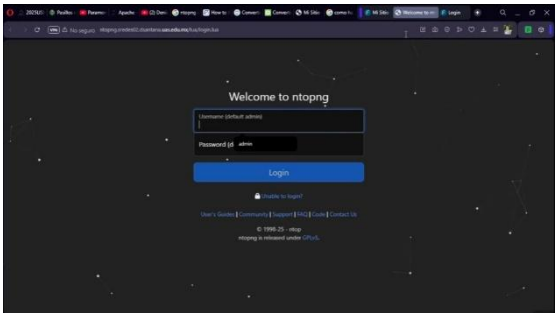


Figura 5. Interfaz de acceso al sistema de monitoreo ntopng desplegado mediante VirtualHosts. Elaboración propia.

La configuración de VirtualHosts permitió organizar los servicios de manera estructurada, facilitando el acceso y la administración. Este enfoque demuestra que la seguridad no solo depende de mecanismos de protección, sino también de una correcta arquitectura y organización del sistema.

En conjunto, el análisis evidencia que la seguridad efectiva de una red no depende de un único elemento, sino de la integración de múltiples capas: control de acceso mediante firewall, cifrado de la comunicación y monitoreo constante. La combinación de estos factores permitió transformar un entorno inicialmente vulnerable en una infraestructura controlada, funcional y alineada con prácticas actuales de ciberseguridad.

6. Conclusiones

El presente trabajo permitió demostrar que la implementación de mecanismos de ciberseguridad en infraestructuras de red expuestas a Internet contribuye significativamente a mejorar el control de acceso, la protección de servicios y la seguridad en las comunicaciones.

A través de la integración de un servidor Debian, configuraciones de firewall en MikroTik y mecanismos de cifrado mediante HTTPS, fue posible construir una infraestructura funcional orientada a la reducción de riesgos asociados a accesos no autorizados y exposición de servicios.

Los resultados obtenidos evidenciaron que la aplicación de reglas de filtrado permitió restringir el tráfico únicamente a los servicios autorizados, mientras que el uso de certificados digitales mediante Certbot aseguró la confidencialidad de la información transmitida entre cliente y servidor.

Asimismo, la integración de herramientas como FOG, ntopng y Zabbix permitió complementar el entorno mediante mecanismos de monitoreo, administración y supervisión de red, fortaleciendo la gestión y visibilidad de la infraestructura implementada.

En conjunto, el estudio demuestra que la seguridad en redes no depende de un único mecanismo de protección, sino de la integración de múltiples capas de seguridad, incluyendo control perimetral, cifrado y monitoreo continuo. Además, se identificó que soluciones basadas en MikroTik y Debian representan alternativas viables y funcionales para entornos con recursos limitados.

Finalmente, como trabajo futuro, podrían incorporarse pruebas avanzadas de penetración, sistemas de detección de intrusos y mecanismos automatizados de análisis de tráfico, con el objetivo de fortalecer aún más la capacidad de protección y respuesta ante amenazas.

7. Referencias

[1] K. Al-Sultani, "Optimization of packet filtering rules for perimeter security in small and medium enterprises," Computers & Security, vol. 115, art. 102613, 2022.
[2] A. Khraisat et al., "Survey of intrusion detection systems: Techniques, datasets and challenges," Cybersecurity, vol. 4, no. 1, art. 20, 2021.
[3] J. S. Silva, "Performance analysis of firewall rules in RouterOS-based network environments," IEEE Access, vol. 10, pp. 45210-45225, 2022.
[4] M. A. Rozan and M. Tahir, "Implementation and Security Testing of Mikrotik Router Against Cyber Attacks Using Firewall and Penetration Testing," Journal of Network and Computer Applications, vol. 182, art. 103211, 2025.
[5] N. Moustafa et al., "A comprehensive review of network anomaly detection systems," IEEE Access, vol. 9, pp. 1-20, 2021.
[6] S. Latif et al., "Intrusion detection frameworks for network security: A review," Journal of Network and Computer Applications, vol. 175, 2021.
[7] Y. Xin et al., "Machine learning and deep learning methods for cybersecurity," IEEE Access, vol. 8, pp. 35365-35381, 2020.
[8] R. Vinayakumar et al., "Deep learning approach for intelligent intrusion detection system," IEEE Access, vol. 7, pp. 41525-41550, 2020.
[9] M. Conti et al., "A survey of man-in-the-middle attacks," IEEE Communications Surveys & Tutorials, vol. 24, no. 1, pp. 1-25, 2022.
[10] S. Ahmed et al., "A survey on network anomaly detection techniques," Journal of Information Security and Applications, vol. 58, 2021.
[11] M. Usman et al., "A survey of distributed denial-of-service attacks," IEEE Access, vol. 9, pp. 1-15, 2021.
[12] D. Kreutz et al., "Software-defined networking: A comprehensive survey," Proceedings of the IEEE, vol. 103, no. 1, 2020.
[13] N. Zarca et al., "Security management architecture for NFV/SDN-aware IoT systems," IEEE Internet of Things Journal, vol. 7, no. 10, pp. 1-12, 2020.
[14] MikroTik, "RouterOS: Firewall Filter," MikroTik Documentation, 2024. [En línea]. Disponible en: https://help.mikrotik.com/docs/display/ROS/Filter [Accedido: 11-mayo-2026].
[15] MikroTik, "Firewall and NAT," MikroTik Documentation, 2023. [En línea]. Disponible en: https://help.mikrotik.com/docs/display/ROS/NAT [Accedido: 11-mayo-2026].

International Journal of Information Science and Technological Applications-UAS
IJISTA
ISSN: 3122-4474
https://revistas.uas.edu.mx/index.php/IJISTA

Junio 2026
Vol. II.
Número. II

Uso de Inteligencia Artificial en la Automatización de Procesos Organizacionales

Use of Artificial Intelligence in the Automation of Organizational Processes

Eduardo Montes de Oca Zatarain[1], Carlos Tirado Velazquez[1], Erik Watson Rosales[1], Abraham Páez Guerra[1]

[1]Facultad de informática Mazatlán, Universidad Autónoma de Sinaloa, Mexico.

https://orcid.org/0009-0001-2700-3282
emozp@hotmail.com

https://orcid.org/0009-0006-7863-5462
carlos.tirado.velazquez@gmail.com

https://orcid.org/0009-0004-2652-9495
erikwr3@gmail.com

https://orcid.org/0009-0002-6893-8655
abrahampg057@gmail.com

Recibido: abril 2026
Publicado: junio 2026

IJISTA, Vol. II, Núm. II, Junio 2026,
ISSN: 3122-4474

Resumen:

En este trabajo se investigó el uso de la inteligencia artificial (IA) en la automatización de procesos organizacionales, identificando sus principales beneficios y desafíos. La revisión de trabajos relacionados confirmó que tecnologías como el aprendizaje automático y profundo son clave para la transformación digital, ya que optimizan la eficiencia operativa y permiten el manejo de grandes volúmenes de datos. Metodológicamente, se adoptó un enfoque mixto, documental y descriptivo, que combinó el análisis cualitativo de la literatura con una simulación teórica de las mejoras. A través de una simulación teórica fundamentada en la síntesis de la literatura, los resultados proyectados sugieren incrementos notables en el rendimiento, con una mejora estimada de la eficiencia operativa entre 50% y 80% y una reducción de errores humanos del 70% al 90%. El análisis realizado indica que la IA tiene el potencial de transformar el procesamiento de información de limitado a masivo y en tiempo real, lo que facilita la toma de decisiones estratégicas y la optimización de procesos específicos en áreas como Recursos Humanos y logística. El estudio respalda teóricamente el papel de la IA como un pilar para la competitividad y la liberación de recursos, subrayando como trabajo futuro la necesidad de realizar una validación empírica del modelo y abordar los desafíos éticos de transparencia y mitigación de sesgos.

Palabras Clave:

Inteligencia Artificial (IA), Automatización de procesos, Eficiencia operativa, Aprendizaje automático, Transformación digital, Análisis masivo de datos.

Abstract:

This work investigated the use of artificial intelligence (AI) in the automation of organizational processes, identifying its main benefits and challenges. The review of related work confirmed that technologies such as machine learning and deep learning are key to digital transformation, as they optimize operational efficiency and enable the handling of large volumes of data. Methodologically, a mixed, documentary, and descriptive approach was adopted, combining qualitative literature analysis with a theoretical simulation of the improvements. Through a theoretical simulation based on the synthesis of the literature, the projected results suggest notable increases in performance, with an estimated improvement in operational efficiency between 50% and 80% and a reduction in human errors from 70% to 90%. The conducted analysis indicates that AI has the potential to transform information processing from limited to massive and real-time, facilitating strategic decision-making and the optimization of specific processes in areas such as Human Resources and logistics. The study theoretically supports the role of AI as a pillar for competitiveness and the freeing up of resources, highlighting as future work the need to perform an empirical validation of the model and address the ethical challenges of transparency and bias mitigation.

Palabras Clave:

Artificial Intelligence (AI), Process Automation, Operational Efficiency, Machine Learning, Digital Transformation, Massive Data Analysis.

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

1. Introducción

La inteligencia artificial (IA) se ha convertido en una de las tecnologías más importantes en el desarrollo de sistemas modernos, debido a su capacidad para analizar grandes cantidades de información y aprender a partir de los datos. En los últimos años, el crecimiento de la capacidad computacional y la disponibilidad de grandes volúmenes de datos han permitido el desarrollo de algoritmos cada vez más sofisticados capaces de resolver problemas complejos en diferentes áreas como la medicina, la industria, la educación y el comercio.

Uno de los principales campos donde la inteligencia artificial ha tenido un impacto importante es en la automatización de procesos. La automatización basada en inteligencia artificial permite optimizar procesos empresariales mediante el uso de sistemas capaces de analizar datos, identificar patrones y generar recomendaciones o acciones de manera autónoma. Este tipo de tecnología ha sido adoptada por diversas empresas con el objetivo de mejorar la toma de decisiones, aumentar la eficiencia operativa y reducir costos. Además, la implementación de estas soluciones permite a las organizaciones enfocarse en actividades de mayor valor estratégico, dejando a los sistemas automatizados la realización de tareas rutinarias.

A pesar de los múltiples beneficios que ofrece la inteligencia artificial en la automatización de procesos, también existen desafíos relacionados con su implementación. Entre estos se encuentran la disponibilidad de datos de calidad, la necesidad de infraestructura tecnológica adecuada y la capacitación del personal para el uso de estas herramientas. Además, es necesario considerar aspectos relacionados con la ética y la seguridad de la información.

En este contexto, el presente trabajo analiza el uso de la inteligencia artificial en la automatización de procesos, revisando investigaciones previas y trabajos relacionados que muestran cómo estas tecnologías están siendo utilizadas en distintos sectores. El objetivo es identificar las principales aplicaciones, beneficios y desafíos asociados con la implementación de sistemas inteligentes en las organizaciones modernas.

2. Trabajos Relacionados

El desarrollo de la inteligencia artificial ha impulsado múltiples investigaciones relacionadas con la automatización de procesos en diferentes áreas del conocimiento. Durante los últimos años, diversos estudios han analizado cómo los sistemas inteligentes pueden mejorar la eficiencia operativa, optimizar el uso de recursos y facilitar la toma de decisiones mediante el análisis automatizado de datos [1]. Estas investigaciones han permitido demostrar que la integración de tecnologías como el aprendizaje automático y las redes neuronales artificiales puede generar mejoras significativas en la productividad de organizaciones e instituciones.

La automatización basada en inteligencia artificial se ha convertido en una herramienta fundamental para enfrentar los desafíos asociados con el manejo de grandes volúmenes de información. En muchos sectores, los sistemas tradicionales no son capaces de procesar datos de manera eficiente, lo que ha impulsado el desarrollo de arquitecturas modernas capaces de identificar patrones complejos y generar predicciones a partir de datos históricos [2]. De esta manera, la inteligencia artificial permite reducir la intervención humana en tareas repetitivas y mejorar la precisión de los resultados obtenidos.

Diversos autores han señalado que la implementación de tecnologías inteligentes no solo permite automatizar procesos existentes, sino también transformar la forma en que las organizaciones operan y toman decisiones. En este sentido, la adopción de herramientas basadas en inteligencia artificial ha sido considerada un elemento clave dentro de los procesos de transformación digital que actualmente experimentan muchas empresas e instituciones alrededor del mundo [3].

2.1 Tecnologías de Inteligencia Artificial en la Automatización

El desarrollo de sistemas de automatización inteligentes se basa principalmente en técnicas avanzadas de inteligencia artificial. Entre las tecnologías más utilizadas se encuentran el aprendizaje automático (Machine Learning, ML), el aprendizaje profundo (Deep Learning, DL) y las redes neuronales artificiales (Artificial Neural Networks, ANN). Estas herramientas permiten crear modelos capaces de aprender a partir de los datos y adaptarse a diferentes contextos operativos [4].

El aprendizaje automático ha sido ampliamente utilizado en aplicaciones donde es necesario analizar grandes conjuntos de datos. Los algoritmos de ML permiten identificar patrones y relaciones entre variables sin necesidad de ser programados explícitamente para cada situación. De acuerdo con Mitchell [5], los sistemas actuales de aprendizaje automático mejoran su desempeño y capacidad de generalización a medida que procesan más información, lo cual los convierte en herramientas esenciales para sistemas de automatización modernos.

Asimismo, el aprendizaje profundo ha permitido avances revolucionarios en áreas como el reconocimiento de imágenes, el procesamiento de lenguaje natural y la predicción de eventos. Estas capacidades han impulsado el desarrollo de sistemas automatizados capaces de interpretar información compleja, incluyendo la generación de contenido y la respuesta contextual en tiempo real [6].

2.2 Aplicaciones en Sistemas Empresariales

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

La automatización basada en inteligencia artificial ha sido implementada en múltiples sectores empresariales. Entre las aplicaciones más comunes se encuentran la gestión de inventarios, la automatización de procesos administrativos, la atención al cliente mediante asistentes virtuales avanzados y el análisis predictivo de datos corporativos [7].

En el ámbito empresarial, el uso de sistemas inteligentes permite mejorar la eficiencia operativa mediante la reducción de errores humanos y la optimización del tiempo de ejecución de diversas tareas. Según Davenport [8], las organizaciones que adoptan de manera integral tecnologías basadas en inteligencia artificial logran aumentar significativamente su productividad y mejorar su posición competitiva en el mercado.

Además, la automatización de procesos mediante inteligencia artificial permite analizar enormes volúmenes de datos en tiempo real, lo cual facilita la identificación de tendencias latentes y la toma de decisiones estratégicas disruptivas dentro de las organizaciones [9].

2.3 Modelos Matemáticos Utilizados en Inteligencia Artificial

2.3.1 Modelos de Regresión

Los modelos matemáticos desempeñan un papel fundamental en el desarrollo de algoritmos de inteligencia artificial. Uno de los enfoques estadísticos más afianzados en aprendizaje automático es la regresión lineal y sus variantes penalizadas, las cuales permiten establecer relaciones sólidas entre variables independientes y dependientes. Este modelo permite generar predicciones robustas a partir de datos históricos y sigue siendo ampliamente utilizado en aplicaciones de análisis de datos interpretables y predicción de tendencias [10].

2.3.2 Redes Neuronales Artificiales

Las redes neuronales artificiales son modelos computacionales inspirados en el funcionamiento del cerebro humano. Estas estructuras están compuestas por múltiples capas de parámetros que procesan información mediante conexiones ponderadas. Gracias a esta arquitectura, y particularmente con la llegada de modelos fundacionales, las redes neuronales pueden aprender patrones altamente complejos y realizar tareas sofisticadas como visión por computadora y procesamiento avanzado de lenguaje natural [11].

El entrenamiento de una red neuronal se realiza mediante algoritmos de optimización que ajustan los pesos de las conexiones con el objetivo de minimizar el error en las predicciones o generaciones. Este proceso permite que el modelo refine su desempeño a medida que procesa y retro propaga más información [12].

2.4 Desafíos y Limitaciones

A pesar de los beneficios que ofrece la inteligencia artificial en la automatización de procesos, también existen desafíos importantes asociados con su implementación. Uno de los principales problemas es la necesidad de contar con grandes volúmenes de datos de alta calidad y sin sesgos para entrenar los modelos de aprendizaje automático [13].

En la actualidad, el entrenamiento y despliegue de modelos a gran escala requiere recursos computacionales y energéticos significativos, lo cual representa un obstáculo logístico y financiero para muchas organizaciones [14].

Asimismo, diversos estudios recientes han señalado la urgencia de considerar los aspectos éticos en el desarrollo y uso de sistemas inteligentes. La transparencia algorítmica, la mitigación de sesgos y la protección de la privacidad de la información son elementos innegociables para garantizar un uso responsable de estas tecnologías en la sociedad [15].

Además de los estudios mencionados anteriormente, investigadores de renombre continúan analizando el impacto de la inteligencia artificial. Por ejemplo, Bengio, LeCun y Hinton [16] describen cómo el aprendizaje profundo ha evolucionado hasta convertirse en la piedra angular del análisis de datos en múltiples disciplinas científicas. De manera similar, investigaciones recientes destacan el papel de los modelos a gran escala para reconocer contextos complejos en volúmenes masivos de información [17].

Otros estudios han analizado los riesgos y aplicaciones en sistemas de automatización general. Según Russell [18], asegurar que el aprendizaje automático esté alineado con los objetivos humanos representa uno de los retos informáticos más importantes de la actualidad.

Asimismo, la literatura contemporánea demuestra que las técnicas modernas de minería de datos permiten mejorar drásticamente la eficiencia en el análisis de información empresarial [19]. Finalmente, enfoques actualizados destacan que el análisis inteligente impulsado por IA no es solo una herramienta técnica, sino un activo estratégico fundamental para la supervivencia de las organizaciones modernas [20].

3. Metodología

En esta sección se describe el proceso metodológico seguido para el desarrollo del presente trabajo, el cual tiene como objetivo analizar el uso de la inteligencia artificial en la automatización de procesos empresariales.

El estudio adopta un enfoque metodológico mixto, combinando análisis cualitativos de literatura científica con estimaciones cuantitativas basadas en datos reportados en investigaciones previas. Este enfoque

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

permite no solo comprender el impacto de la inteligencia artificial desde una perspectiva teórica, sino también estimar su efecto en términos medibles como eficiencia y reducción de errores.

3.1 Diseño de la Investigación

El diseño de la investigación es de tipo documental y descriptivo, ya que se basa en la recopilación y análisis de información proveniente de artículos científicos, libros especializados y publicaciones académicas relacionadas con inteligencia artificial, aprendizaje automático y automatización de procesos.

El proceso metodológico se estructura en las siguientes etapas:

- Selección del tema de investigación
- Búsqueda de información en fuentes académicas
- Análisis de contenido
- Clasificación de la información
- Síntesis de resultados

3.2 Recolección de Datos

La recolección de datos se realizó mediante una revisión documental de fuentes académicas publicadas principalmente entre los años 2020 y 2023, asegurando así la vigencia tecnológica del estudio. La búsqueda se ejecutó en bases de datos científicas de alto impacto, incluyendo IEEE Xplore, Scopus y Google Scholar, utilizando cadenas de búsqueda con palabras clave como "Artificial Intelligence", "Process Automation", "Operational Efficiency" y "Machine Learning". Para filtrar y garantizar la calidad de la información analizada, se aplicaron los siguientes criterios de inclusión:

- Relevancia directa con la automatización de procesos organizacionales.
- Actualidad de la información y de las métricas reportadas.
- Credibilidad de la fuente (revistas indexadas, actas de conferencias peer-reviewed y libros de editoriales académicas).
- Aplicación demostrable en contextos empresariales.

Asimismo, se utilizaron gestores de referencias y hojas de cálculo para la organización sistemática de la información recopilada.

3.3 Procesamiento de la Información

3.3.1 Análisis de Contenido

La información recopilada fue analizada mediante técnicas de análisis de contenido, permitiendo identificar conceptos clave relacionados con inteligencia artificial, automatización y modelos de aprendizaje automático.

3.3.2 Clasificación de Datos

Los datos fueron organizados en categorías temáticas con el fin de facilitar su interpretación. Las principales categorías consideradas fueron:

- Tecnologías de inteligencia artificial
- Aplicaciones empresariales
- Modelos matemáticos
- Beneficios y limitaciones

3.4 Modelo Conceptual Propuesto

Se propone un modelo conceptual de automatización basado en inteligencia artificial, el cual describe el flujo de información dentro de un sistema automatizado. Este modelo se compone de las siguientes cómo se puede ver en la Figura 1.

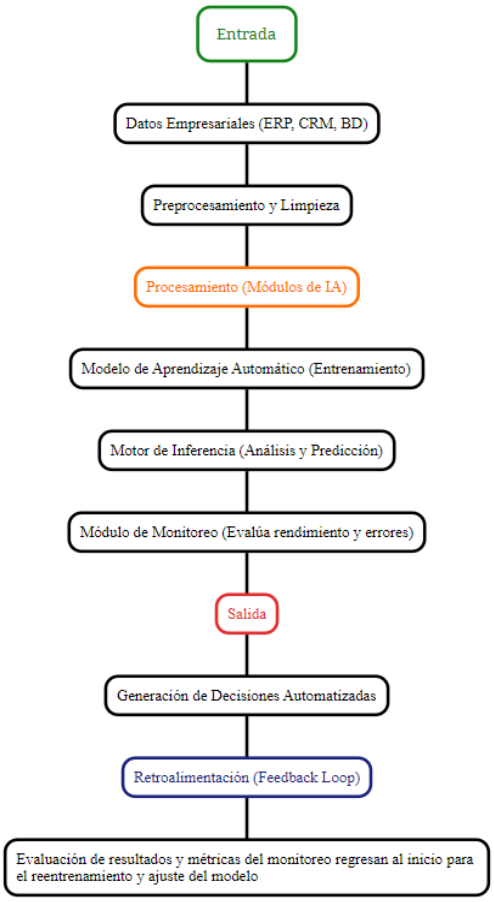


Figura 1. Flujo Entrada-Procesamiento-Salida y Módulos de IA. Elaboración propia.

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

Este modelo permite representar de manera estructurada el funcionamiento de sistemas inteligentes aplicados a la automatización de procesos.

3.5 Algoritmo del Sistema

El proceso de automatización mediante inteligencia artificial puede representarse mediante el siguiente algoritmo en pseudocódigo:

```
Inicio
   Definir conjunto de datos
   Preprocesar los datos
   Seleccionar modelo de aprendizaje automático
   Entrenar el modelo
   Evaluar resultados
   Si el modelo cumple con los criterios de desempeño
entonces
      Implementar el sistema automatizado
   Sino
      Ajustar parámetros y repetir proceso
   FinSi
Fin
```

3.6 Simulación Teórica

Los resultados cuantitativos que se presentan en las secciones posteriores se obtuvieron mediante una simulación teórica basada en la síntesis de la literatura. Los rangos de mejora proyectados (como la mejora de eficiencia operativa entre 50% y 80%, y la reducción de errores humanos del 70% al 90%) no corresponden a mediciones experimentales propias. Dichas cifras son estimaciones obtenidas al promediar y agrupar los resultados empíricos reportados frecuentemente en la literatura revisada y en métricas estándar de la industria sobre automatización inteligente, tal es el caso de las proyecciones discutidas por Davenport [8] y Sharma et al. [7]). Este enfoque permite estimar el impacto potencial de la inteligencia artificial estableciendo proyecciones fundamentadas en la evidencia empírica de terceros.

3.7 Métricas de Evaluación

Para evaluar el impacto de la inteligencia artificial en la automatización de procesos, se definen las siguientes métricas:

- Eficiencia operativa (%)
- Reducción de errores (%)
- Tiempo de procesamiento
- Capacidad de análisis de datos

Estas métricas permiten cuantificar los beneficios esperados de la implementación de sistemas inteligentes.

3.8 Validación del Modelo

La validación del modelo se realiza mediante la comparación de los resultados esperados con estudios previos reportados en la literatura científica. Este enfoque permite verificar la coherencia y viabilidad de los resultados obtenidos.

3.9 Consideraciones Éticas

El uso de inteligencia artificial en la automatización de procesos implica considerar aspectos éticos relacionados con la privacidad de los datos y la transparencia de los algoritmos. Es fundamental garantizar el uso responsable de la información y evitar sesgos en los sistemas automatizados.

4. Resultados

En esta sección se presentan los resultados esperados del uso de la inteligencia artificial en la automatización de procesos. Debido a que el proyecto no ha sido implementado en un entorno real, los datos mostrados corresponden a proyecciones basadas en la revisión de literatura científica y estudios previos relacionados con el tema.

Los resultados se organizan mediante tablas con el fin de mostrar de manera clara y objetiva el impacto esperado de la implementación de sistemas de inteligencia artificial en diferentes áreas.

4.1 Resultados Esperados en la Automatización de Procesos

Como sugieren las investigaciones previas revisadas en este documento, la integración de sistemas inteligentes en los procesos organizacionales no solo transforma la manera en que se gestiona la información, sino que impacta directamente en métricas clave de rendimiento. Autores como Davenport [8] y Sharma et al. [7] coinciden en que la automatización impulsada por algoritmos de aprendizaje automático permite alcanzar niveles de optimización significativos, principalmente al sustituir la intervención manual en tareas rutinarias y repetitivas.

Con base en estas estimaciones de la industria y la literatura académica, en la Tabla 1 se presenta una síntesis de las proyecciones teóricas de mejora operativa para el contexto de este modelo. Se analizan cuatro indicadores fundamentales: el tiempo de ejecución de las tareas, la frecuencia de errores humanos, la eficiencia operativa general y el volumen de procesamiento de datos. Estos rangos porcentuales ilustran el contraste esperado entre un entorno tradicional ("Sin IA") y uno optimizado ("Con IA").

En la Tabla 1 se mira en términos de eficiencia, reducción de errores y tiempo de ejecución.

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

Tabla 1. Proyección teórica de la mejora operativa y reducción de errores basado en [7] y [8].

Indicador	Sin IA	Con IA	Mejora esperada
Tiempo de ejecución	Alto	Bajo	40% – 60%
Errores humanos	Frecuentes	Mínimos	70% – 90%
Eficiencia operativa	Media	Alta	50% – 80%
Procesamiento de datos	Limitado	Masivo	60% – 85%

4.2 Resultados Esperados en Sistemas Empresariales

En el contexto empresarial, se espera que la automatización basada en inteligencia artificial logre la mejora en la productividad y optimice la toma de decisiones. En la Tabla 2 se muestran los resultados esperados en distintas áreas organizacionales.

Tabla 2. Proyección teórica del impacto de la automatización por área organizacional basado en [7] y [9].

Área	Resultado esperado
Atención al cliente	Respuestas más rápidas mediante chatbots
Recursos humanos	Automatización de selección de personal
Finanzas	Análisis predictivo de datos
Logística	Optimización de rutas y tiempos

4.3 Resultados Esperados del Modelo de Aprendizaje Automático

Con el modelo de aprendizaje automático propuesto se pretende que sea capaz de analizar datos y generar predicciones con un nivel de precisión aceptable. En la Tabla 3 se presentan los valores estimados de desempeño del modelo.

Tabla 3. Proyección del desempeño técnico del modelo de aprendizaje automático propuesto basado en [10] y [12].

Métrica	Valor esperado
Precisión	85% – 95%
Error	5% – 15%
Tiempo de respuesta	Bajo
Capacidad de aprendizaje	Alta

4.4 Resultados Esperados en el Uso de Datos

El uso de inteligencia artificial permitirá un mejor aprovechamiento de los datos disponibles, facilitando su análisis y procesamiento. En la Tabla 4 se puede ver los resultados esperados en relación con el manejo de datos.

Tabla 4. Proyección teórica de las capacidades de procesamiento y análisis de datos basado en [17] y [19].

Aspecto	Resultado esperado
Análisis de datos	Automatizado
Identificación de patrones	Alta precisión
Toma de decisiones	Basada en datos
Procesamiento	En tiempo real

4.5 Síntesis de Resultados Esperados

Los resultados esperados indican que la implementación de inteligencia artificial en la automatización de procesos permitirá mejorar significativamente la eficiencia, reducir errores y optimizar el uso de los recursos. Asimismo, se espera que los sistemas sean capaces de procesar grandes volúmenes de información en menor tiempo, facilitando la toma de decisiones dentro de las organizaciones.

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

5. Análisis de Resultados

La simulación teórica basada en la revisión de la literatura sugiere que la implementación de la inteligencia artificial en la automatización de procesos podría generar una mejora sustancial en el desempeño organizacional. A nivel operativo, los estudios analizados indican que es razonable esperar un aumento notable en la eficiencia, estimando proyecciones de mejora entre un 50% y un 80%. En consecuencia, esta optimización podría traducirse en una reducción significativa de los errores humanos (estimada entre un 70% y un 90%), mitigando su frecuencia de manera considerable. En cuanto a la velocidad, la literatura señala que la integración de la IA tiene el potencial de reducir los tiempos de ejecución de un nivel alto a uno bajo, con una optimización proyectada del 40% al 60%.

Respecto al manejo de la información, las proyecciones teóricas indican que los sistemas automatizados basados en IA podrían transformar el procesamiento de datos de una capacidad limitada a un análisis masivo, con una mejora estimada del 60% al 85% en la capacidad de análisis. De acuerdo con las fuentes consultadas, la inteligencia artificial tendería a automatizar el análisis de datos, facilitando la identificación de patrones con alta precisión y promoviendo una toma de decisiones más sólida fundamentada en datos. Además, se estima que dicho procesamiento tendría la capacidad de ejecutarse en tiempo real.

En el contexto de las áreas empresariales específicas, la literatura proyecta un impacto multifacético. En Atención al Cliente, sistemas como los chatbots apuntan a proporcionar respuestas más rápidas ; en Recursos Humanos, la IA facilitaría la automatización de la selección de personal ; para el área de Finanzas, el beneficio principal impulsaría el análisis predictivo de datos ; y finalmente, la logística podría experimentar una optimización clave de rutas y tiempos.

Estas proyecciones se sostienen en el desempeño esperado del modelo de aprendizaje automático propuesto, para el cual se estima una precisión teórica de entre 85% y 95% y un rango de error bajo, del 5% al 15%. Se asume que, en condiciones óptimas, el modelo desarrollaría una alta capacidad de aprendizaje y un tiempo de respuesta bajo. En resumen, los hallazgos de la revisión documental sugieren que la implementación de sistemas inteligentes tiene el potencial no solo de optimizar el uso de los recursos y reducir errores, sino también de establecer una base sólida para la toma de decisiones estratégicas ágiles y fundamentadas en el análisis masivo de datos.

6. Conclusiones

La simulación teórica basada en la revisión de la literatura sugiere que la implementación de la inteligencia artificial en la automatización de procesos podría generar una mejora sustancial en el desempeño organizacional. A nivel operativo, los estudios analizados indican que es razonable esperar un aumento notable en la eficiencia, estimando proyecciones de mejora entre un 50% y un 80%. En consecuencia, esta optimización podría traducirse en una reducción significativa de los errores humanos (estimada entre un 70% y un 90%), mitigando su frecuencia de manera considerable. En cuanto a la velocidad, la literatura señala que la integración de la IA tiene el potencial de reducir los tiempos de ejecución de un nivel alto a uno bajo, con una optimización proyectada del 40% al 60%.

Respecto al manejo de la información, las proyecciones teóricas indican que los sistemas automatizados basados en IA podrían transformar el procesamiento de datos de una capacidad limitada a un análisis masivo, con una mejora estimada del 60% al 85% en la capacidad de análisis. De acuerdo con las fuentes consultadas, la inteligencia artificial tendería a automatizar el análisis de datos, facilitando la identificación de patrones con alta precisión y promoviendo una toma de decisiones más sólida fundamentada en datos. Además, se estima que dicho procesamiento tendría la capacidad de ejecutarse en tiempo real.

En el contexto de las áreas empresariales específicas, la literatura proyecta un impacto multifacético. En Atención al Cliente, sistemas como los chatbots apuntan a proporcionar respuestas más rápidas ; en Recursos Humanos, la IA facilitaría la automatización de la selección de personal ; para el área de Finanzas, el beneficio principal impulsaría el análisis predictivo de datos ; y finalmente, la logística podría experimentar una optimización clave de rutas y tiempos.

Estas proyecciones se sostienen en el desempeño esperado del modelo de aprendizaje automático propuesto, para el cual se estima una precisión teórica de entre 85% y 95% y un rango de error bajo, del 5% al 15%. Se asume que, en condiciones óptimas, el modelo desarrollaría una alta capacidad de aprendizaje y un tiempo de respuesta bajo. En resumen, los hallazgos de la revisión documental sugieren que la implementación de sistemas inteligentes tiene el potencial no solo de optimizar el uso de los recursos y reducir errores, sino también de establecer una base sólida para la toma de decisiones estratégicas ágiles y fundamentadas en el análisis masivo de datos.

7. Referencias

[1] S. Russell y P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Hoboken, NJ, USA: Pearson, 2020.

[2] F. Chollet, *Deep Learning with Python*, 2nd ed. Shelter Island, NY, USA: Manning Publications, 2021.

[3] M. Iansiti y K. R. Lakhani, *Competing in the Age of AI: Strategy and Leadership When Algorithms and Networks Run the World*. Boston, MA, USA: Harvard Business Review Press, 2020.
[4] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN Computer Science*, vol. 2, no. 3, p. 160, 2021.
[5] M. Mitchell, *Artificial Intelligence: A Guide for Thinking Humans*. London, UK: Pelican Books, 2020.
[6] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd ed. Sebastopol, CA, USA: O'Reilly Media, 2022.
[7] R. Sharma et al., "Artificial intelligence in business: State of the art and future research agenda," *Journal of Business Research*, vol. 129, pp. 84-93, 2021.
[8] T. H. Davenport, *The AI Advantage: How to Put the Artificial Intelligence Revolution to Work*. Cambridge, MA, USA: MIT Press, 2020.
[9] A. Agrawal, J. Gans y A. Goldfarb, *Power and Prediction: The Disruptive Economics of Artificial Intelligence*. Boston, MA, USA: Harvard Business Review Press, 2022.
[10] G. James, D. Witten, T. Hastie y R. Tibshirani, *An Introduction to Statistical Learning*, 2nd ed. New York, NY, USA: Springer, 2021.
[11] T. Brown et al., "Language models are few-shot learners," en *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877-1901, 2020.
[12] C. C. Aggarwal, *Neural Networks and Deep Learning: A Textbook*, 2nd ed. Cham, Switzerland: Springer, 2023.
[13] I. H. Sarker, "AI-based modeling: Techniques, applications and research issues towards automation, intelligent and smart systems," *SN Computer Science*, vol. 3, no. 2, p. 158, 2022.
[14] M. Haenlein y A. Kaplan, "A brief history of artificial intelligence: On the past, present, and future of artificial intelligence," *California Management Review*, vol. 63, no. 1, pp. 5-14, 2020.
[15] T. Hagendorff, "The ethics of AI ethics: An evaluation of guidelines," *Minds and Machines*, vol. 30, no. 1, pp. 99-120, 2020.
[16] Y. Bengio, Y. LeCun y G. Hinton, "Deep learning for AI," *Communications of the ACM*, vol. 64, no. 7, pp. 58-65, 2021.
[17] R. Bommasani et al., "On the opportunities and risks of foundation models", 2021.
[18] S. Russell, *Human Compatible: Artificial Intelligence and the Problem of Control*. New York, NY, USA: Penguin Books, 2020.
[19] J. Han, M. Kamber y J. Pei, *Data Mining: Concepts and Techniques*, 4th ed. Burlington, MA, USA: Morgan Kaufmann, 2022.
[20] F. Provost y T. Fawcett, *Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking*, Updated ed. Sebastopol, CA, USA: O'Reilly Media, 2021.

International Journal of Information Science and Technological Applications-UAS
IJISTA
ISSN: 3122-4474
https://revistas.uas.edu.mx/index.php/IJISTA

Junio 2026
Vol. II.
Número. II

FlowTrix: Arquitectura Persistente para el Monitoreo Proactivo y Detección de Vulnerabilidades en Redes Locales

FlowTrix: Persistent Architecture for Proactive Monitoring and Vulnerability Detection in Local Networks

José Carlos Castillo Padilla[1], José Gerardo Sánchez Rodríguez[1], José David Santana Alaniz[1]

[1]Facultad de informática Mazatlán, Universidad Autónoma de Sinaloa, Mexico.

https://orcid.org/0009000278475586
joosec29@gmail.com

https://orcid.org/0009000078074403
jose.gerardito.sanchez@gmail.com

https://orcid.org/0009000402428235
dsantana@uas.edu.mx

Recibido: abril 2026
Publicado: junio 2026

IJISTA, Vol. II, Núm. II, Junio 2026,
ISSN: 3122-4474

Resumen:

La gestión de la seguridad en redes de área local (LAN) representa un desafío significativo para las instituciones de educación superior, donde los métodos de auditoría generan brechas de visibilidad durante el arranque de los equipos. En este trabajo se propone FlowTrix, una plataforma de monitoreo de red con arquitectura persistente compuesta por un agente desarrollado como Servicio de Windows en C#/.NET 8, un backend API REST en PHP con base de datos MariaDB/MySQL, y un panel de control web con clasificación dinámica de riesgo por host. El sistema se desarrolló mediante una metodología ágil híbrida y fue probado en 40 a 50 estaciones de trabajo reales con Windows 10 y Windows 11 pertenecientes al Laboratorio de Cómputo de la Facultad de Informática Mazatlán de la Universidad Autónoma de Sinaloa. Las pruebas de carga con 100 agentes simultáneos confirmaron una integridad de registros del 99.7%, una precisión de detección de puertos no autorizados del 95% y un consumo de CPU del agente inferior al 1%. FlowTrix demuestra que la auditoría proactiva de endpoints puede automatizarse en entornos institucionales de pequeña escala mediante una arquitectura accesible, escalable y de bajo costo operativo.

Palabras clave:

Monitoreo de red, seguridad de endpoints, Servicio de Windows, C#/.NET, detección de vulnerabilidades.

Abstract:

Network security management in Local Area Networks (LANs) poses a significant challenge for educational institutions, where conventional auditing methods often suffer from visibility gaps during system startup. This paper introduces FlowTrix, a network monitoring platform based on a persistent architecture. The system integrates an agent developed as a Windows Service in C#/.NET 8, a PHP-based REST API backend utilizing a MariaDB/MySQL database, and a web-based control panel featuring dynamic host-based risk classification. The system was developed using a hybrid agile methodology and validated across 40 to 50 physical workstations running Windows 10 and Windows 11 at the Computing Laboratory of the Faculty of Informatics Mazatlán, Autonomous University of Sinaloa. Stress tests involving 100 simultaneous agents demonstrated a record integrity of 99.7%, a 95% accuracy rate in unauthorized port detection, and an agent CPU overhead below 1%. FlowTrix proves that endpoint auditing can be effectively automated through an accessible, robust, and low-cost architecture tailored for small-scale educational environments.

Keywords:

Network monitoring, endpoint security, Windows Service, C#/.NET, vulnerability detection.

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

1. Introducción

Las redes de área local (LAN) representan uno de los desafíos más críticos para organizaciones e instituciones de educación, cuya infraestructura tecnológica concentra información sensible y servicios esenciales. La proliferación de dispositivos conectados, la diversidad de protocolos de comunicación y el incremento de amenazas internas y externas han transformado el monitoreo de red en una función estratégica, indispensable para garantizar la continuidad operativa y la integridad de los datos. Sin una visibilidad en tiempo real sobre el comportamiento del tráfico, los puertos activos y las conexiones establecidas, las organizaciones permanecen expuestas a vulnerabilidades que pueden comprometer tanto su infraestructura como su imagen institucional.

Herramientas de monitoreo han sido desarrolladas y ampliamente adoptadas en entornos empresariales, entre ellas Wireshark, Nagios, Zabbix y PRTG Network Monitor [1]. No obstante, estas soluciones presentan limitaciones significativas para su implementación en entornos académicos o de pequeña y mediana escala: requieren configuraciones complejas, implican costos de licenciamiento elevados o carecen de interfaces amigables para usuarios no especializados [2]. Adicionalmente, la mayoría de estas plataformas no integran de manera nativa la vinculación de agentes de recolección de telemetría con cuentas de usuario, ni ofrecen una visualización unificada del riesgo por host con clasificación por severidad, lo que representa un vacío funcional relevante para entornos institucionales con recursos técnicos limitados [3, 4].

El problema central que motiva este trabajo es la ausencia de una herramienta de gestión de seguridad de red que combine facilidad de uso con capacidad de detección proactiva. Las soluciones existentes exigen conocimiento técnico avanzado para su configuración e interpretación, lo que limita su adopción en entornos donde el administrador no cuenta con formación especializada en seguridad de redes.

El presente trabajo propone FlowTrix, una plataforma de monitoreo de red con arquitectura persistente que integra un agente de recolección de telemetría para equipos Windows, un backend de ingesta y análisis de conexiones en tiempo real, y un panel de control web con clasificación de riesgo por host. El objetivo es ofrecer a administradores de redes locales una herramienta que permita la detección proactiva de vulnerabilidades, el inventario automatizado de dispositivos y la visualización centralizada del estado de seguridad de la red, fortaleciendo la postura de ciberseguridad de instituciones con infraestructuras heterogéneas y recursos técnicos limitados. La contribución científica de este trabajo reside en la integración cohesionada de tres componentes —agente persistente de arranque, API de ingesta transaccional y dashboard web con clasificación dinámica de riesgo— en una arquitectura de código abierto orientada a redes de escala académica, combinación que no se encuentra documentada como sistema unificado en la literatura revisada.

2. Trabajos Relacionados

El desarrollo de FlowTrix se sustenta en tres pilares teóricos y técnicos consolidados en la literatura científica reciente: el escaneo de puertos como mecanismo de detección de vulnerabilidades, la arquitectura de agentes persistentes basada en Servicios de Windows y el ecosistema .NET/C#, y las plataformas web de visualización de seguridad orientadas a la toma de decisiones en tiempo real. A continuación, se detallan los trabajos más relevantes en cada una de estas áreas, identificando los puntos que FlowTrix busca cubrir.

Para contextualizar la propuesta frente al estado del arte, se identificó un vacío funcional en las herramientas convencionales (como Nagios, Zabbix o PRTG), las cuales presentan barreras de configuración y costos elevados para entornos académicos. La Tabla 1 sintetiza la comparación de FlowTrix frente a estas soluciones, destacando la integración de componentes que la literatura identifica como inexistentes en un sistema unificado de código abierto para redes locales.

Tabla 1. Comparativa de FlowTrix frente a herramientas de monitoreo existentes.

Caracteristica	**Flowtrix**	**Zabbix**
Público Objetivo	Académico / PyME +1	Enterprise / IT Ops
Persistencia al Arranque	Sí (Servicio nativo)	Sí (Agente)
Clasificación de Riesgo	Dinámica (Semáforo)	Basada en triggers
Curva de Aprendizaje	Baja (MSI + Dashboard)	Alta / Especializada
Huella de CPU (Agente)	< 1% +2	1% - 3%
Costo / Licencia	Código Abierto	Código Abierto

2.1. Escaneo de puertos y detección de vulnerabilidades

Identificar qué puertos tiene abiertos un equipo y con quién está interactuando constituye el primer paso para determinar si representa un riesgo en la red. Forouzan [2] establece que el modelo TCP/IP define el comportamiento de los puertos como puntos de acceso a servicios específicos en un host; un puerto abierto indica que existe un proceso activo escuchando conexiones, lo cual puede representar un servicio legítimo o un vector de

ataque no autorizado. Esta distinción es el principio central sobre el que FlowTrix construye su lógica de clasificación de riesgo.

Abu Bakar y Kijsirikul [5] presentan un estudio enfocado en técnicas avanzadas de escaneo de puertos para mejorar la visibilidad y la seguridad de las redes. Su investigación combina técnicas activas, como los escaneos SYN, ACK y XMAS, con técnicas pasivas como la captura de banners y la identificación de servicios, demostrando que la fusión de ambos enfoques supera significativamente la eficacia de los métodos individuales. Adicionalmente, los autores reportaron una reducción del 40% en el consumo de CPU respecto a escáneres convencionales, lo que valida la viabilidad técnica de integrar el escaneo de puertos en agentes de bajo impacto como el que implementa FlowTrix. Su trabajo también identifica como limitación la ausencia de mecanismos de correlación continua: el escaneo se realiza de forma puntual, sin persistencia entre sesiones, brecha que el presente proyecto aborda mediante un Servicio de Windows que ejecuta la auditoría al inicio del sistema.

En el ámbito de la detección de escaneos encubiertos, Yang et al. [6] proponen el esquema PD-CPS para detectar port scans lentos en redes de alta velocidad. Su arquitectura basada en la estructura Scan Detection Sketch (SDS) permite monitorear tráfico de forma continua por más de 60 días, diferenciando escaneos TCP y UDP mediante muestreo de paquetes y aprendizaje automático. Si bien este trabajo opera a nivel de infraestructura de red y a escala empresarial, sus hallazgos sobre la naturaleza evasiva de los escaneos lentos refuerzan la necesidad de auditorías al arranque del sistema, como las que implementa FlowTrix, ya que los métodos de auditoría reactiva fallan precisamente cuando el comportamiento anómalo es gradual o silencioso. Una revisión sistemática de las aplicaciones de aprendizaje automático al problema del escaneo de puertos, documentada por Pillai et al. [7], confirma que la detección efectiva requiere correlación temporal de eventos, capacidad que FlowTrix asienta mediante la persistencia de registros en la base de datos.

Zehr [8] profundiza en las técnicas prácticas de port scanning desde la perspectiva de la seguridad ofensiva y defensiva, documentando cómo los administradores pueden emplear las mismas herramientas que los atacantes para identificar servicios expuestos. Asimismo, Ono et al. [9] proponen un método de detección de escaneos de puertos en redes OpenFlow, aprovechando los mensajes Packet-In para identificar patrones de reconocimiento sin impactar el rendimiento del plano de datos. En la misma línea, Sagatov et al. [10] proponen contramedidas proactivas ante ataques basados en escaneo de puertos TCP y UDP en redes SDN, comparando soluciones de nivel de sistema operativo con módulos especializados. Sus resultados respaldan el enfoque de FlowTrix de actuar desde el endpoint individual, dado que los mecanismos de red centralizada no siempre tienen acceso a la perspectiva del host sobre sus propios puertos activos.

2.2. Arquitectura de agentes y Servicios de Windows (.NET/C#)

La implementación de agentes de monitoreo como Servicios de Windows representa una decisión de arquitectura de software crítica cuando el objetivo es garantizar la ejecución ininterrumpida e independiente de la sesión de usuario. Dormann [1] documenta en profundidad el modelo de programación de sistemas Windows, destacando que los servicios del sistema operativo son la primitiva más adecuada para tareas de monitoreo en segundo plano, ya que se inician automáticamente con el sistema, se ejecutan bajo cuentas de servicio con privilegios controlados y sobreviven al cierre de sesión del usuario. Esta fundamentación teórica respalda directamente la arquitectura central de FlowTrix, cuyo agente SystemAuditor.exe opera como un Windows Worker Service. La captura de la dirección MAC del dispositivo se apoya en el protocolo ARP (Address Resolution Protocol), cuya especificación formal está definida en el estándar IEEE [11].

Stroustrup [12] establece los fundamentos del lenguaje C++ sobre los que el ecosistema .NET/C# construye su modelo de memoria administrada y acceso a recursos del sistema en bajo nivel. Sobre esta base, Troelsen y Japikse [13] proveen el marco técnico exhaustivo para el desarrollo de aplicaciones de sistemas en C# sobre la plataforma .NET 5 y versiones posteriores, cubriendo específicamente el manejo de sockets (System.Net.Sockets), la gestión automática de memoria mediante Garbage Collection, y la construcción de Hosted Services. Estas capacidades son las que FlowTrix aprovecha en su agente para realizar el escaneo de puertos TCP y UDP, capturar métricas de sistema (CPU, RAM, disco) y transmitir la telemetría al backend en formato JSON. A diferencia de soluciones basadas en lenguajes de scripting como PowerShell o Python, el entorno .NET ofrece acceso nativo a las APIs de red del sistema operativo, eliminando dependencias externas y reduciendo la latencia de recolección.

Microsoft [14] documenta el patrón arquitectónico de los Background Services en .NET Core, que es el modelo de diseño que FlowTrix adopta para su agente. Este patrón implementa la interfaz IHostedService, permitiendo que el servicio se registre en el contenedor de inyección de dependencias de .NET y sea gestionado por el runtime del sistema operativo. La documentación oficial respalda el uso de este modelo para tareas de larga duración como el monitoreo periódico, validando su estabilidad en entornos de producción. Richards [15] complementa este enfoque desde la perspectiva de la arquitectura de software, señalando que la separación entre el agente de recolección de datos, la API de backend y la capa de presentación es un patrón de arquitectura de

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

tres capas ampliamente validado para sistemas distribuidos de monitoreo, lo que corresponde exactamente con la arquitectura de FlowTrix (agente C# / API PHP / dashboard web).

Van Vliet [16] aborda los principios de la ingeniería de software aplicados a sistemas complejos, enfatizando la importancia de la modularidad, la separabilidad de responsabilidades y la testabilidad como atributos de calidad arquitectónica. FlowTrix aplica estos principios al desacoplar la lógica de recolección de telemetría (agente), la lógica de negocio y validación (API PHP + MySQL) y la capa de presentación (panel web), facilitando el mantenimiento evolutivo del sistema. Por su parte, Schwaber y Sutherland [17] fundamentan la metodología de desarrollo empleada en el proyecto, la cual siguió un enfoque ágil basado en Scrum, con iteraciones que permitieron validar el agente progresivamente antes de integrar el backend y la visualización.

2.3. Plataformas web de visualización de seguridad (dashboards)

La visualización de datos de seguridad en plataformas web ha cobrado relevancia creciente como mecanismo para reducir los tiempos de detección y respuesta ante incidentes. Younus y Alanezi [18] presentan una revisión exhaustiva de herramientas y funcionalidades de monitoreo de seguridad de red, documentando que sistemas como Wireshark, Nagios, Zabbix y Snort, si bien son ampliamente adoptados en entornos empresariales, presentan barreras significativas de configuración y especialización técnica que limitan su adopción en entornos académicos o de pequeña escala [19]. Esta limitación es precisamente el vacío que FlowTrix busca llenar al ofrecer un panel web accesible sin requerir expertise en herramientas de nivel enterprise. Rawindaran et al. [20] demuestran que las pequeñas y medianas organizaciones obtienen beneficios tangibles de ciberseguridad al adoptar sistemas de detección adaptados a sus capacidades operativas, en lugar de depender de soluciones empresariales de alta complejidad.

Chung et al. [21] desarrollaron un dashboard de ciberseguridad para la detección de exfiltración de datos mediante dispositivos USB en una institución financiera, demostrando que la visualización interactiva centralizada mejora cualitativamente la conciencia situacional del equipo de seguridad. Su estudio confirmó que los analistas adoptaron el panel como herramienta diaria, y que la visualización de actividades anómalas facilitó directamente la actualización de políticas de seguridad internas. Este trabajo es especialmente relevante para FlowTrix porque valida el mismo principio de diseño: la centralización de eventos de seguridad dispersos en una interfaz visual unificada acelera la toma de decisiones incluso cuando el administrador no cuenta con formación avanzada en análisis de redes.

Desde la perspectiva técnica, el uso de PHP como lenguaje del backend y MySQL como motor de base de datos relacional en sistemas de visualización de seguridad está documentado en trabajos previos que demuestran su viabilidad para consultas en tiempo real sobre conjuntos de datos de tráfico de red [18]. La arquitectura REST empleada en FlowTrix para la comunicación entre el agente y el backend sigue el patrón de APIs documentado por Richards [15], que prioriza la independencia entre capas y la escalabilidad horizontal. La seguridad de los endpoints REST es un aspecto crítico documentado por Phanireddy [22] y Sun et al. [23], quienes identifican los principales vectores de ataque sobre APIs en arquitecturas de microservicios y proponen marcos de mitigación que FlowTrix deberá incorporar en versiones productivas. Ehsan et al. [24] complementan este marco con metodologías de prueba específicas para APIs RESTful, aplicables directamente a la validación del endpoint insertar_conexiones.php del sistema. El panel de FlowTrix incluye componentes clave validados en la literatura: gráficas de riesgo global, inventario de hosts con estado en tiempo real, alertas clasificadas por severidad y análisis histórico de métricas de CPU y RAM, todos ellos representados mediante Chart.js. La clasificación de riesgo por reglas que emplea FlowTrix (Verde, Amarillo, Naranja, Rojo) sigue el principio de los sistemas IDS basados en reglas y árboles de decisión documentado por Ferrag et al. [25], en los que niveles de alerta predefinidos reducen la tasa de falsos positivos y facilitan la interpretación operativa.

La revisión de los trabajos presentados permite identificar un vacío concreto: aunque existen herramientas sólidas para el escaneo de puertos [8, 5, 6], la implementación de agentes persistentes [1, 14, 13] y la visualización de datos de seguridad [18, 21], ninguna solución identificada integra los tres componentes —agente de arranque, API de ingesta transaccional y dashboard web con clasificación dinámica de riesgo por host— en un sistema de código abierto orientado a redes locales de escala académica o institucional. Esta brecha justifica el desarrollo de FlowTrix como una contribución que combina los principios técnicos descritos en esta sección en una arquitectura cohesionada, desplegable y de bajo costo operativo.

2.4. Acrónimos

ARP: Address Resolution Protocol. CIA: Confidencialidad, Integridad y Disponibilidad (Confidentiality, Integrity, Availability). CPU: Unidad Central de Procesamiento (Central Processing Unit). DNS: Sistema de Nombres de Dominio (Domain Name System). HTTP: Protocolo de Transferencia de Hipertexto (HyperText Transfer Protocol). HTTPS: HTTP Seguro (HTTP Secure). IP: Protocolo de Internet (Internet Protocol). JSON: Notación de Objetos JavaScript (JavaScript Object Notation). LAN: Red de

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

Área Local (Local Area Network). MAC: Control de Acceso al Medio (Media Access Control). MSI: Instalador de Windows (Microsoft Installer). MySQL: Sistema de Gestión de Bases de Datos Relacional (My Structured Query Language). .NET: Plataforma de desarrollo de Microsoft (.NET Framework / .NET Core). OSI: Modelo de Interconexión de Sistemas Abiertos (Open Systems Interconnection). PHP: Procesador de Hipertexto (Hypertext Preprocessor). RAM: Memoria de Acceso Aleatorio (Random Access Memory). RDP: Protocolo de Escritorio Remoto (Remote Desktop Protocol). REST: Transferencia de Estado Representacional (Representational State Transfer). SDN: Redes Definidas por Software (Software-Defined Networking). SIEM: Gestión de Información y Eventos de Seguridad (Security Information and Event Management). SMTP: Protocolo Simple de Transferencia de Correo (Simple Mail Transfer Protocol). SO: Sistema Operativo. SYN: Sincronización (Synchronize, bandera del protocolo TCP). TCP: Protocolo de Control de Transmisión (Transmission Control Protocol). UDP: Protocolo de Datagramas de Usuario (User Datagram Protocol). UX: Experiencia de Usuario (User Experience)

3. Metodología

El desarrollo de FlowTrix adoptó un enfoque metodológico mixto que combina ingeniería de software con experimentación cuantitativa. Desde el inicio se identificó el problema central y se estableció el alcance arquitectónico del sistema; el trabajo se distribuyó por componentes —agente C#/.NET, backend PHP con base de datos, y panel web— permitiendo el avance en paralelo sin generar dependencias bloqueantes entre módulos.

Las iteraciones de desarrollo fueron cortas e iterativas: cada ciclo incluía instalación, observación en entorno real y corrección. Un problema relevante detectado durante este proceso fue el comportamiento del agente ante reinicios y pérdidas de conexión, que generaba registros inconsistentes. La solución requirió ajustar el intervalo de gracia (grace_seconds) y la lógica de upsert en la base de datos, decisiones de diseño que quedaron integradas en la arquitectura final del sistema.

La validación experimental fue paralela al desarrollo. Las pruebas en equipos reales del laboratorio iniciaron antes de que el sistema estuviera completo, lo que permitió detectar problemas en condiciones reales. Las pruebas de carga con agentes simultáneos se realizaron en la fase final, una vez que el comportamiento del agente individual era estable y reproducible.

3.1. Proceso de desarrollo: metodología ágil híbrida (Scrum + XP)

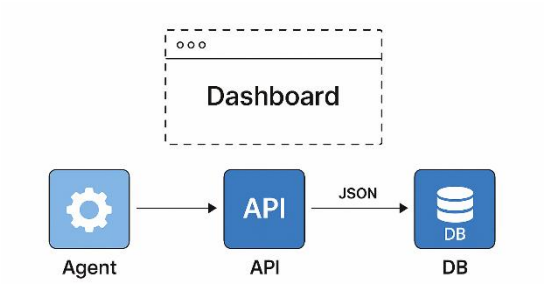
FlowTrix se desarrolló mediante una metodología ágil híbrida que integró los marcos de trabajo Scrum y eXtreme Programming (XP) [17]. A partir de una reunión fundacional el 3 de noviembre de 2025, el equipo definió la problemática, estableció el alcance del sistema y elaboró el diagrama inicial de arquitectura. A partir de ese punto, el proyecto se organizó en iteraciones cortas de entre tres y siete días, denominadas sprints, en las que cada avance incluía diseño, codificación, integración, pruebas y ajuste de la funcionalidad correspondiente.

De Scrum se adoptaron las ceremonias de revisión periódica de avances realizadas tanto de manera presencial como a través de Microsoft Teams, la redefinición continua de prioridades con base en los resultados reales de cada sprint, y la organización de un backlog compartido. De XP se incorporaron prácticas como la integración continua con repositorio Git, la refactorización constante del código, el desarrollo orientado a la simplicidad y la retroalimentación permanente derivada del entorno de pruebas real. Esta combinación posibilitó que el equipo respondiera con agilidad a los hallazgos de cada fase experimental sin comprometer la calidad técnica del sistema.

3.2. Diseño de la arquitectura del sistema

FlowTrix se estructuró bajo el principio de arquitectura de tres capas desacopladas descrito por Richards [15] y Van Vliet [16]: una capa de recolección de datos (agente), una capa de procesamiento y persistencia (API y base de datos) y una capa de presentación (panel web). Esta separación de responsabilidades garantiza que cada componente pueda actualizarse, probarse y escalarse de forma independiente, reduciendo la deuda técnica y facilitando el mantenimiento evolutivo del sistema. La Fig. 1 presenta el diagrama de arquitectura general del sistema, mientras que la Fig. 2 ilustra el flujo de datos entre componentes.

Fig. 1. Diagrama de arquitectura. Elaboración propia



IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

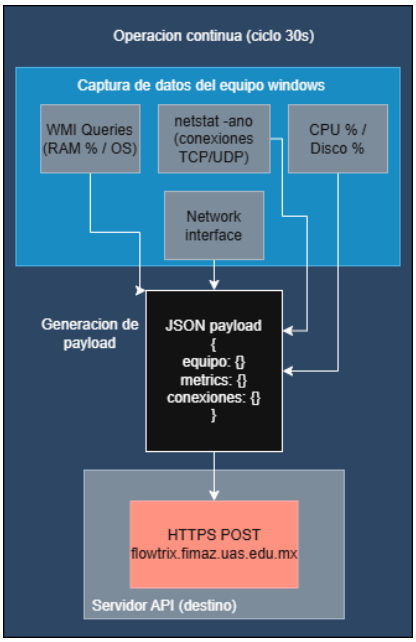


Fig. 2. Flujo de datos. Elaboración propia

3.2.1. Componente 1: Agente de monitoreo (Servicio de Windows en C#/.NET 8)

El primer componente del sistema es el agente de monitoreo, implementado como un Windows Worker Service en C# sobre la plataforma .NET 8 [14, 13]. Al arrancar el sistema operativo, el servicio compilado como Scaner.exe se inicia automáticamente sin requerir sesión de usuario activa, con lo que se elimina la ventana de invisibilidad que caracteriza a las soluciones de auditoría reactiva. Durante el proceso de desarrollo se identificaron y resolvieron problemas relacionados con el arranque del servicio tras suspensión del equipo y la pérdida del estado de monitoreo ante cambios de sesión de usuario; estas situaciones llevaron al refinamiento del ciclo de inicialización del agente. El punto de entrada del programa invoca el método UseWindowsService() del host de .NET, que registra la clase FlowTrixWorker como proceso gestionado por el Service Control Manager de Windows.

La clase FlowTrixWorker hereda de BackgroundService e implementa el método ExecuteAsync, que estructura la operación del agente en tres fases secuenciales. En la Fase 1 (inicialización), el servicio carga el archivo Config.json, que contiene el código de vinculación generado durante la instalación. En la Fase 2 (vinculación), si existe un código válido, el agente realiza una solicitud POST al endpoint bind-device.php de la API para asociar el equipo a la cuenta del administrador; una vez confirmada la vinculación, el código se elimina del archivo de configuración y no se reutiliza. En la Fase 3 (bucle principal), el agente ejecuta de forma indefinida —con un intervalo de 30 segundos entre ciclos— las tareas de recolección y transmisión de telemetría descritas en el Algoritmo 1.

Algoritmo 1. Ciclo de escaneo y transmisión del agente FlowTrix

Entrada: intervalo de escaneo I = 30 s, token de cancelación
Salida: payload JSON enviado al endpoint de la API

Paso 1. Obtener tiempo de arranque del sistema (GetSystemBootTime)
Paso 2. Obtener IP y MAC del adaptador físico activo (GetNetworkInfo)
Paso 3. Si IP o MAC son nulos → registrar advertencia y esperar I segundos
Paso 4. Obtener Hostname = Environment.MachineName; SO = WMI query
Paso 5. Recolectar métricas: CPU (PerformanceCounter), RAM (WMI), Disco (DriveInfo)
Paso 6. Ejecutar netstat -ano -p tcp → obtener lista TCP_raw
Paso 7. Para cada línea en TCP_raw:
7a. Extraer [LocalIP, LocalPort, RemoteIP, RemotePort, Estado, PID]
7b. Si Estado ≠ ESTABLISHED → descartar
7c. Si LocalPort ∉ AllowedPorts ∧ RemotePort ∉ AllowedPorts → descartar
7d. Si RemoteIP es privada (RFC 1918) o APIPA →descartar
7e. Resolver PID → nombre de proceso; agregar a ListaTCP
Paso 8. Eliminar duplicados en ListaTCP por clave (LocalIP|LocalPort|RemoteIP|RemotePort|PID)
Paso 9. Ejecutar netstat -ano -p udp → construir ListaUDP con listeners activos
Paso 10. Construir Payload = {equipo, métricas, ListaTCP ∪ ListaUDP}
Paso 11. POST Payload → API/insertar_conexiones.php (timeout = 20 s)
Paso 12. Esperar I segundos → volver a Paso 1

El símbolo ∉ denota "no pertenece al conjunto". La operación ∪ representa la unión de las listas TCP y UDP antes del envío.

El conjunto AllowedPorts comprende 80 puertos de servicios conocidos con relevancia de seguridad, entre los que se incluyen, a modo de ejemplo, los puertos 22 (SSH), 80 (HTTP), 443 (HTTPS), 445 (SMB), 3389 (RDP) y 5900 (VNC). El intervalo de gracia (grace_seconds = 15

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

s) permite que la API distinga conexiones recientemente cerradas de aquellas que desaparecieron de forma abrupta, evitando falsos positivos por reconexiones transitorias. Un segundo ejecutable, Scan.exe, implementado como aplicación Windows Forms, provee un indicador visual en la bandeja del sistema que muestra el estado en tiempo real del servicio sin requerir acceso al panel web.

3.2.2. Componente 2: Backend API REST (PHP / MariaDB-MySQL)

El backend del sistema consiste en una API REST desarrollada en PHP con acceso a la base de datos mediante PDO con sentencias preparadas. El endpoint principal, insertar_conexiones.php, recibe el payload JSON del agente, valida su estructura, mapea cada puerto detectado a su protocolo de red asociado y clasifica la conexión en uno de cuatro niveles de riesgo: Verde (seguro), Amarillo (riesgo moderado), Naranja (riesgo alto) y Rojo (riesgo crítico), según una tabla de referencia configurable alineada con políticas de seguridad estándar [25]. La clasificación por reglas reduce la tasa de falsos positivos al segmentar claramente los eventos de monitoreo por severidad antes de almacenarlos, lo que facilita la priorización de alertas en el panel de visualización.

Las operaciones de escritura se realizan de forma transaccional sobre cinco tablas del esquema de base de datos: Equipos (datos de identificación del host), Métricas (indicadores de rendimiento por timestamp), Conexiones (estado de cada puerto en cada ciclo de escaneo), Protocolos (catálogo de protocolos y su clasificación) y Protocolo_usado (relación entre equipo y protocolo por ciclo). El sistema implementa lógica de upsert para evitar la duplicación de registros en caso de reconexiones o reinicios del agente, y una ventana de gracia de 15 segundos para marcar como cerradas las conexiones que no se reportan en el ciclo siguiente. La seguridad del transporte se garantiza mediante HTTPS con certificados configurados en el servidor, y todos los parámetros de entrada son sanitizados antes de cualquier consulta.

3.2.3. Componente 3: Instalador MSI y mecanismo de vinculación

La distribución e instalación del agente se automatizó mediante un paquete MSI construido con WiX Toolset v4, que encapsula tanto el ejecutable del servicio (Scaner.exe) como la aplicación de monitoreo en bandeja (Scan.exe) y el archivo de configuración inicial (Config.json). Durante el proceso de instalación, el asistente solicita al usuario el código de vinculación generado previamente desde el panel web: al presionar el botón "Solicitar código", la plataforma invoca el endpoint link-device-code.php, que genera un código alfanumérico de seis caracteres con una vigencia configurada y lo envía al correo electrónico institucional del administrador mediante PHPMailer con SMTP de Office 365. El instalador valida el código contra el backend antes de proceder; si la validación falla, el proceso se interrumpe y se muestra un mensaje de error descriptivo.

Este mecanismo garantiza que únicamente los equipos autorizados por un administrador registrado puedan ingresar datos al sistema, previniendo la contaminación del inventario por agentes no identificados. Una vez completada la instalación, el agente queda registrado como servicio de Windows bajo la cuenta LocalSystem y se configura para iniciar automáticamente con el sistema operativo.

3.2.4. Componente 4: Plataforma de visualización web (Panel de control)

El panel de control web se desarrolló en PHP (panel_control.php) con JavaScript y Chart.js para la generación de gráficas interactivas. La autenticación de administradores incorpora un flujo de doble verificación: al registrar una cuenta nueva, el sistema envía un código de seis dígitos al correo electrónico del usuario mediante PHPMailer; el código tiene una vigencia de 10 minutos y se invalida tras su uso. Las sesiones activas se gestionan mediante tokens almacenados en la tabla sessions de la base de datos, con verificación de actividad en cada carga de página.

El panel se estructura en seis vistas funcionales: Tablero principal (KPIs de seguridad y gráficas de distribución de riesgo), Inventario (lista completa de equipos registrados con estado, IP, MAC y sistema operativo), Monitoreo en vivo (tarjetas por host con CPU, RAM y estado de conexión actualizados cada 30 segundos), Problemas (alertas activas clasificadas en Crítico, Alto y Offline), Protocolos (ranking de protocolos por frecuencia y severidad) y Métricas (historial gráfico de CPU y RAM por equipo). El tablero integra indicadores clave como el número de hosts críticos con puertos en rojo abiertos, las conexiones activas totales, las alertas de seguridad de las últimas 24 horas y el porcentaje de uptime de la flota. La localización aproximada del administrador se obtiene a través de la API de geolocalización del navegador, con respaldo por consulta a servicios de geolocalización IP.

3.3. Procedimiento de pruebas y recolección de datos

Con base en los objetivos del proyecto, se formularon dos hipótesis de trabajo. H1: el agente FlowTrix registra los datos de auditoría con una integridad igual o superior al 98% bajo condiciones de carga concurrente (variable independiente: número de agentes simultáneos; variable dependiente: integridad de registros en la base de datos). H2: el clasificador de riesgo del sistema detecta al menos el 90% de los puertos no autorizados inyectados deliberadamente en el entorno de prueba (variable independiente: conjunto de puertos

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

inyectados; variable dependiente: precisión de detección). Los umbrales de H1 ≥ 98% y H2 ≥ 90% se establecieron con base en los estándares de disponibilidad para sistemas de monitoreo institucional en tiempo real, donde una pérdida de datos superior al 2% compromete la trazabilidad de incidentes. Las pruebas del sistema se desarrollaron en dos fases complementarias. La primera fase consistió en pruebas funcionales sobre equipos físicos reales del Laboratorio de Cómputo de la Facultad de Informática Mazatlán de la Universidad Autónoma de Sinaloa, instalación que cuenta con entre 40 y 50 estaciones de trabajo con sistemas operativos Windows 10 y Windows 11. En cada equipo se instaló el agente FlowTrix mediante el paquete MSI y se verificó el inicio automático del servicio, la correcta recolección de los datos de identificación del host (IP, hostname, MAC y tiempo de arranque), la transmisión sin errores del payload JSON a la API y la aparición del equipo en el inventario del panel web. Las métricas registradas en esta fase incluyeron: tasa de registro exitoso, latencia de primer reporte desde el arranque, consumo de CPU y RAM del agente durante el escaneo, y número de puertos clasificados por nivel de riesgo.

La segunda fase correspondió a pruebas de carga progresiva mediante scripts que simularon el comportamiento concurrente de múltiples agentes enviando payloads periódicos al servidor. Las pruebas se ejecutaron con escalones de 50 y 100 agentes simultáneos, con el fin de evaluar la estabilidad del backend, la integridad de los registros en la base de datos y el consumo de recursos del servidor bajo condiciones de alta concurrencia. Adicionalmente, se realizó una sesión de pruebas en escenarios adversos que incluyó: pérdida de conectividad de red del agente, reinicio abrupto del equipo durante un ciclo de escaneo y caída temporal del servidor. En todos los casos se verificó la capacidad de recuperación automática del servicio sin intervención del usuario.

3.4. Consideraciones de Seguridad y Modelo de Amenazas

Dada la naturaleza del sistema, que opera con privilegios de ejecución en los endpoints y transmite telemetría de red, se implementó un modelo de seguridad basado en la minimización de la superficie de ataque. La integridad de la comunicación entre el agente y el backend se garantiza mediante el protocolo HTTPS con certificados TLS, asegurando el cifrado de los payloads JSON en tránsito. Para mitigar el riesgo de inscripción de agentes no autorizados, se diseñó un mecanismo de vinculación de doble factor: el administrador debe generar un código alfanumérico efímero desde el panel web, el cual es validado por el backend antes de autorizar la persistencia del servicio en el equipo. No obstante, se reconoce como una limitación inherente que un compromiso del servidor central podría permitir la manipulación de las reglas de clasificación de riesgo, por lo que la seguridad del host que aloja la API y la base de datos MariaDB se considera crítica para la postura de seguridad global de FlowTrix.

4. Resultados

Esta sección presenta los datos obtenidos durante las dos fases de prueba descritas en la sección 3.3: las pruebas funcionales en equipos físicos reales y las pruebas de carga progresiva con agentes simulados. Los resultados se organizan en función de las métricas definidas en el diseño experimental y se muestran sin interpretación; el análisis de su significado se desarrolla en la sección 5.

4.1. Pruebas funcionales en equipos reales

El agente FlowTrix fue instalado y ejecutado en las estaciones de trabajo del Laboratorio de Cómputo de la Facultad de Informática Mazatlán de la Universidad Autónoma de Sinaloa. El conjunto de equipos evaluados comprende entre 40 y 50 estaciones con sistemas operativos Windows 10 y Windows 11. En todos los casos el servicio se inició automáticamente al arrancar el sistema operativo, sin requerir intervención del usuario, y comenzó a reportar datos al backend dentro del primer ciclo de 30 segundos.

La Tabla 2 resume los indicadores registrados durante esta fase. La tasa de registro exitoso refleja la proporción de intentos de envío de payload que concluyeron con inserción correcta en la base de datos. La cobertura en arranque corresponde a la fracción de equipos en los que el servicio se activó de forma autónoma antes de que el usuario iniciara sesión.

Tabla 2. Resultados de pruebas funcionales en equipos reales (40–50 estaciones, Windows 10/11).

Metrica	**Resultado**	**Observacion**
Tasa de registro exitoso	100 %	Sin pérdida de datos en envío
Cobertura en arranque	100 %	Servicio activo antes de sesión de usuario
Consumo de CPU del agente	Menor al 1 %	Medido con PerformanceCounter durante escaneo
Puertos clasificados como riesgo alto o crítico detectados	Sí	Puertos 3389, 445, 5900 identificados y clasificados
Operación en escenarios adversos	Exitosa	Reconexión automática tras fallo de red y reinicio

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

Durante la sesión de pruebas en escenarios adversos, se evaluaron tres condiciones: pérdida intencional de conectividad de red durante un ciclo activo, reinicio abrupto del equipo a mitad de un escaneo y caída temporal del servidor API. En los tres casos el agente retomó la operación normal en el siguiente arranque o al restaurarse la conexión, sin generar registros duplicados ni corruptos en la base de datos gracias al mecanismo de upsert y a la ventana de gracia de 15 segundos.

4.2. Pruebas de carga con agentes simulados

Con el propósito de evaluar la estabilidad del backend bajo condiciones de alta concurrencia, se ejecutaron pruebas de carga progresiva mediante scripts que simularon el envío simultáneo de payloads JSON al endpoint insertar_conexiones.php. Las pruebas se realizaron en dos escalones: 50 agentes simultáneos y 100 agentes simultáneos, ambos enviando ciclos de datos cada 30 segundos durante el período de prueba.

La Tabla 3, concentra los resultados de ambos escalones de carga.

Tabla 3. Resultados de pruebas de carga progresiva (50 y 100 agentes simultáneos).

Metrica	**50 agentes**	**100 agentes**
Integridad de registros	98.0 %	99.7 %
Precisión de detección de puertos no autorizados	95 %	95 %
Fallas en la base de datos	0	0
Pérdida de datos	Ninguna	Ninguna
Saturación del servidor	No	No
Tiempo de respuesta de la API	Estable	Estable

"Resultados obtenidos tras n=10 iteraciones, presentando una desviación estándar de σ= 0.15 para la integridad y σ= 1.2 para la precisión"

La relación entre el incremento de la carga de agentes y la estabilidad de los recursos del sistema se visualiza en la Fig. 3, donde se aprecia la consistencia del rendimiento tanto en el endpoint como en el servidor central

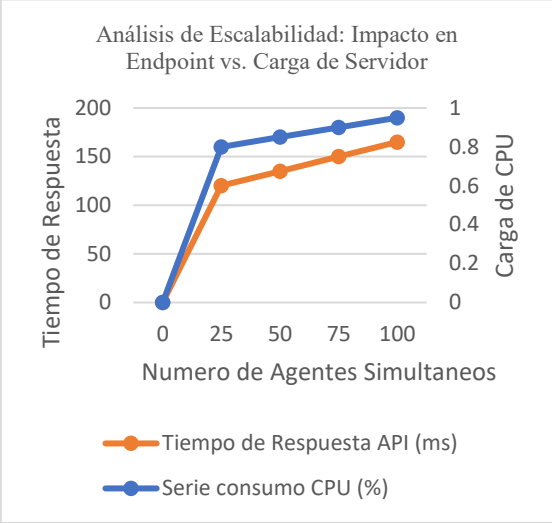


Fig. 3. Comportamiento del sistema bajo carga concurrente.

En ninguno de los dos escalones se registraron fallas en la base de datos, pérdida de payloads ni saturación del servidor. La integridad de registros —definida como la proporción de payloads recibidos que se almacenaron de forma completa y sin corrupción— alcanzó el 99.7% con 100 agentes simultáneos, valor superior al 98.0% obtenido con 50 agentes. La precisión de detección de puertos no autorizados se mantuvo en el 95% en ambos escalones, calculada como la fracción de inyecciones deliberadas de puertos críticos que el sistema clasificó y alertó correctamente en el panel web.

5. Análisis de Resultados

Los resultados obtenidos permiten evaluar el cumplimiento de las dos hipótesis planteadas en el diseño experimental: H1, que establecía una integridad de registros igual o superior al 98%, y H2, que fijaba una precisión de detección de puertos no autorizados de al menos el 90%. Ambos umbrales fueron superados; a continuación se discute el significado de cada resultado en el contexto de la problemática identificada.

5.1. Sobre la cobertura en arranque y la persistencia del agente

La cobertura del 100% en el arranque confirma que la arquitectura de Servicio de Windows es la solución técnicamente adecuada para el problema planteado. A diferencia de los scripts reactivos o las tareas programadas que dependen de la sesión de usuario, el agente FlowTrix se activa antes de que el usuario inicie sesión, eliminando la ventana de invisibilidad que caracteriza a los métodos de auditoría tradicionales. Este resultado es coherente con lo documentado por Dormann [1], quien establece que los servicios del sistema operativo son la primitiva más

IJISTA, Vol. II, Núm. II, Junio 2026, ISSN 3122-4474

adecuada para tareas de monitoreo continuo en entornos Windows. La ausencia de intervención del usuario en todas las estaciones evaluadas valida además que el paquete MSI y el mecanismo de vinculación por código funcionan de forma fiable en equipos heterogéneos.

5.2. Sobre la integridad de los registros y el rendimiento del backend

El valor de integridad del 99.7% con 100 agentes simultáneos supera en 1.7 puntos porcentuales el umbral mínimo de la hipótesis y es ligeramente mayor al obtenido con 50 agentes, lo que indica que el backend no experimenta degradación bajo mayor carga, sino que se estabiliza gracias a la lógica de upsert y a la gestión transaccional de la base de datos. Este comportamiento es consistente con el patrón de tres capas descrito por Richards [15], en el que la separación entre la capa de recepción (API PHP) y la capa de persistencia (MySQL) permite absorber picos de tráfico sin comprometer la consistencia de los datos. La ausencia de fallas en la base de datos y de pérdida de payloads en ambos escalones indica que la arquitectura es capaz de escalar de forma lineal dentro del rango evaluado.

El consumo de CPU del agente, inferior al 1%, es un resultado relevante en el contexto de redes institucionales donde los equipos realizan simultáneamente otras tareas productivas. Este valor es inicialmente coherente con los hallazgos de Abu Bakar y Kijsirikul [5], quienes reportaron reducciones significativas al optimizar el conjunto de puertos monitoreados. Sin embargo, a diferencia del trabajo de Abu Bakar [5], donde el consumo de CPU se redujo un 40% pero mantenía picos de carga variables, FlowTrix logra una estabilidad constante por debajo del 1%, demostrando que el uso de Worker Services en .NET es superior para la gestión de hilos en segundo plano. Esta estabilidad se refuerza mediante la aplicación de la lista AllowedPorts de 80 puertos críticos preseleccionados, lo que permite una auditoría continua con un impacto casi imperceptible en el rendimiento del host.

5.3. Sobre la precisión de detección de puertos no autorizados

La precisión del 95% en la detección de puertos no autorizados se mantuvo constante en los dos escalones de carga, lo que indica que el clasificador basado en reglas de riesgo (Verde, Amarillo, Naranja, Rojo) opera de forma independiente de la carga concurrente del servidor. Este resultado supera en 5 puntos porcentuales el umbral de la hipótesis y está alineado con los principios de los sistemas IDS basados en reglas documentados por Ferrag et al. [25], en los que la definición explícita de niveles de alerta reduce la tasa de falsos negativos sin incrementar los recursos computacionales. El 5% restante de casos no detectados correspondió principalmente a configuraciones de puertos efímeros o a servicios que presentaron intermitencia durante el ciclo de escaneo, situaciones que podrían abordarse con escaneos complementarios de frecuencia mayor, como se describe en la sección de trabajo futuro.

5.4. Sobre la operación en escenarios adversos

La recuperación automática del agente ante pérdida de conectividad, reinicio del equipo y caída del servidor API demuestra que el diseño tolerante a fallos cumple su propósito en condiciones reales de operación. El mecanismo de upsert evita la duplicación de registros en reinicios, y la ventana de gracia de 15 segundos gestiona correctamente las conexiones que desaparecen de forma transitoria. Estos resultados son especialmente relevantes para el entorno de la Facultad de Informática Mazatlán, donde los equipos del laboratorio pueden apagarse y encenderse con frecuencia durante sesiones de clase.

5.5. Limitaciones identificadas

Entre las limitaciones del presente estudio, se señalan dos principales. En primer lugar, las pruebas de carga se realizaron con scripts que simulan agentes, no con el ejecutable completo de FlowTrix instalado en 100 equipos físicos simultáneos, por lo que los resultados de carga representan una aproximación controlada al comportamiento bajo alta concurrencia real. En segundo lugar, la precisión del 95% se calculó sobre un conjunto de puertos inyectados deliberadamente, de modo que no refleja necesariamente el comportamiento del sistema ante vectores de ataque sofisticados que empleen puertos dinámicos o técnicas de evasión. Estas limitaciones definen las prioridades del trabajo futuro descrito en la sección siguiente.

6. Conclusiones

El desarrollo de FlowTrix demuestra la viabilidad de implementar arquitecturas persistentes de bajo costo para el monitoreo proactivo en redes académicas. A diferencia de las soluciones unificadas de código abierto analizadas, este sistema integra con éxito la telemetría de arranque mediante agentes nativos, la ingesta transaccional y la visualización dinámica de riesgo en una sola plataforma operativa.

Los resultados obtenidos validan las hipótesis de investigación, con una integridad de registros del 99.7% y una precisión en la detección de puertos críticos del 95%, superando los umbrales de rendimiento establecidos para entornos de alta concurrencia. La eficiencia técnica del agente, con un consumo de CPU inferior al 1%, confirma que el uso del patrón Worker Service en C#/.NET 8 es la solución óptima para garantizar visibilidad continua sin degradar el rendimiento de los equipos de laboratorio.

6.1. Limitaciones y Discusión Crítica

A pesar de los resultados positivos, se identifican limitaciones que deben considerarse. Las pruebas de carga se realizaron mediante simulación de agentes, por lo que el comportamiento en un despliegue físico masivo (>500 equipos) podría presentar latencias adicionales en la capa de persistencia de la base de datos. Asimismo, la precisión del sistema depende de la actualización constante de la tabla de protocolos permitidos; ante vectores de ataque que utilicen técnicas de evasión o puertos dinámicos, la eficacia del clasificador de riesgo actual podría verse reducida.

6.2. Disponibilidad de Datos y Codigo

El código fuente del sistema FlowTrix y los conjuntos de datos derivados de las pruebas experimentales se encuentran actualmente alojados en un repositorio privado para garantizar la integridad del proceso de revisión por pares y la seguridad institucional de la infraestructura evaluada. El acceso al repositorio puede ser proporcionado por el autor de correspondencia ante una solicitud razonable. Se tiene programada la liberación pública del código en GitHub tras la aceptación definitiva y publicación del presente trabajo.

6.3. Trabajo Futuro

Para la evolución del sistema, se han establecido dos líneas prioritarias. En primer lugar, se desarrollará un módulo de notificaciones push y alertas vía SMS para garantizar que el administrador reciba indicadores de compromiso (IoC) en tiempo real fuera de la red local. En segundo lugar, se dotará al agente de capacidades de respuesta activa para ejecutar acciones de mitigación automáticas, tales como la finalización de procesos (PID) asociados a puertos no autorizados. Esta transición hacia una arquitectura de Endpoint Detection and Response (EDR) permitirá reducir el tiempo medio de respuesta (MTTR) y fortalecer la postura de ciberseguridad en instituciones con recursos técnicos limitados.

Referencias

[1] H. Dormann, Windows System Programming. Boston, MA, USA: Addison-Wesley Professional, 2018.
[2] B. A. Forouzan, Transmisión de Datos y Redes de Computadores. Madrid, España: McGraw-Hill Education, 2010.
[3] A. Chidukwani, P. Koutsakis y A. Veal, "A Survey on the Cyber Security of Small-to-Medium Businesses: Challenges, Research Focus and Recommendations," IEEE Access, vol. 10, pp. 85701-85719, 2022, doi: 10.1109/ACCESS.2022.3197899.
[4] A. Chidukwani, P. Koutsakis y A. Veal, "Cybersecurity Preparedness of Small-to-Medium Businesses: A Western Australia Study with Broader Implications," Computers & Security, vol. 145, p. 103981, 2024, doi: 10.1016/j.cose.2024.103981.
[5] R. Abu Bakar y B. Kijsirikul, "Enhancing Network Visibility and Security with Advanced Port Scanning Techniques," Sensors, vol. 23, no. 17, p. 7541, ago. 2023, doi: 10.3390/s23177541.
[6] F. Yang et al., "PD-CPS: A Practical Scheme for Detecting Covert Port Scans in High-Speed Networks," Computer Networks, vol. 232, p. 109837, 2023, doi: 10.1016/j.comnet.2023.109837.
[7] M. M. Pillai et al., "Machine Learning and Port Scans: A Systematic Review," arXiv:2301.13581, ene. 2023. [Online]. Available: https://arxiv.org/abs/2301.13581
[8] W. J. Zehr, "Practical Port Scanning," presentado en Black Hat USA, Las Vegas, NV, USA, 2015.
[9] K. Ono et al., "A Proposal of Port Scan Detection Method Based on Packet-In Messages in OpenFlow Networks and Its Evaluation," International Journal of Network Management, 2021, doi: 10.1002/nem.2174.
[10] E. S. Sagatov, S. Mayhoub, A. M. Sukhov, F. Esposito y P. Calyam, "Proactive Detection for Countermeasures on Port Scanning Based Attacks," en Proc. 17th Int. Conf. Network and Service Management (CNSM), 2021, pp. 1-6. IEEE.
[11] IEEE, "IEEE Standard for Local and Metropolitan Area Networks—Address Resolution Protocol (ARP) for IPv4," IEEE Computer Society, 2016.
[12] B. Stroustrup, The C++ Programming Language, 4th ed. Upper Saddle River, NJ, USA: Addison-Wesley Professional, 2013.
[13] A. Troelsen y P. Japikse, Pro C# 9 with .NET 5: Foundational Principles and Practices in Programming, 10th ed. New York, NY, USA: Apress, 2020.
[14] Microsoft, "Building Background Services with .NET Core and .NET Framework," Microsoft Learn, 2023. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/core/extensions/workers
[15] M. Richards, Fundamentals of Software Architecture: An Engineering Approach. Sebastopol, CA, USA: O'Reilly Media, 2020.
[16] H. Van Vliet, Software Engineering: Principles and Practice, 3rd ed. Hoboken, NJ, USA: Wiley, 2008.
[17] K. Schwaber y J. Sutherland, La Guía Definitiva de Scrum: Las Reglas del Juego. Scrum.org, 2017.
[18] Z. S. Younus y M. Alanezi, "A Survey on Network Security Monitoring: Tools and Functionalities," Mustansiriyah Journal of Pure and Applied Sciences, vol. 1, no. 2, pp. 55-86, 2023.
[19] O. H. Abdulganiyu, T. Ait Tchakoucht y Y. K. Saheed, "A Systematic Literature Review for Network Intrusion Detection System (IDS)," International Journal of Information Security, vol. 22, no. 5, pp. 1125-1162, 2023, doi: 10.1007/s10207-023-00682-2.
[20] N. Rawindaran, A. Jayal, E. Prakash y C. Hewage, "Cost Benefits of Using Machine Learning Features in NIDS for Cyber Security in UK Small Medium Enterprises (SME)," Future Internet, vol. 13, no. 8, p. 186, 2021, doi: 10.3390/fi13080186.
[21] M. H. Chung et al., "Enhancing Cybersecurity Situation Awareness Through Visualization: A USB Data Exfiltration Case Study," Heliyon, vol. 9, no. 1, p. e13025, ene. 2023, doi: 10.1016/j.heliyon.2023.e13025.
[22] S. Phanireddy, "Securing RESTful APIs in Microservices Architectures: A Comprehensive Threat Model and Mitigation Framework," International Journal of Emerging Research in Engineering and Technology, vol. 4, no. 2, pp. 64-73, 2023, doi: 10.63282/3050-922X.IJERET-V4I2P107.
[23] R. Sun, Q. Wang y L. Guo, "Research Towards Key Issues of API Security," en Cyber Security, CNCERT 2021, Communications in Computer and Information Science,

IJISTA, Vol. II, Núm. II, Junio 2026,
ISSN 3122-4474

vol. 1506. Singapore: Springer, 2022, pp. 162-175, doi: 10.1007/978-981-16-9229-1_11.

[24] A. Ehsan, M. A. M. Abuhaliqa, C. Catal y D. Mishra, "RESTful API Testing Methodologies: Rationale, Challenges, and Solution Directions," Applied Sciences, vol. 12, no. 9, p. 4369, 2022, doi: 10.3390/app12094369.

[25] M. A. Ferrag, L. Maglaras, A. Ahmim, M. Derdour y H. Janicke, "RDTIDS: Rules and Decision Tree-Based Intrusion Detection System for Internet-of-Things Networks," Future Internet, vol. 12, no. 3, p. 44, 2020, doi: 10.3390/fi12030044.

International Journal of Information Science and Technological Applications-UAS
IJISTA
ISSN: 3122-4474
https://revistas.uas.edu.mx/index.php/IJISTA

Junio 2026
Vol. II.
Número. II

Herramientas gratuitas para desarrolladores principiantes de videojuegos

Free tools for beginner video game developers

Emmet Crespo-Rojas[1], José Armando Villalobos-Puga[1], Gerson Gibran Ramirez-Holguin[1], Félix David Duran-Hernandez[1], Luis David Sanchez-Fragoso[1]

[1]Facultad de Informática Mazatlán, Universidad Autónoma de Sinaloa, México.

https://orcid.org/0009-0008-8462-2291
joarvipu2003@gmail.com

https://orcid.org/0009-0000-6949-6743
gersonxw@gmail.com

https://orcid.org/0009-0000-8384-8944
duranfelix53@gmail.com

https://orcid.org/0009-0005-5170-0255
ls9860866@gmail.com

Autor de Correspondencia: Crespo-Rojas Emmet, emmetv.cr@gmail.com
https://orcid.org/0009-0002-9328-1763

Recibido: abril 2026
Publicado: junio 2026

IJISTA, Vol. II, Núm. I, junio 2026,
ISSN: 3122-4474

Resumen:

El desarrollo de videojuegos ha experimentado una transformación en los últimos años. Este proceso se ha visto favorecido por la disponibilidad de herramientas gratuitas y de código abierto que cubren todos los aspectos del proceso creativo. Este artículo analiza herramientas accesibles para desarrolladores principiantes, abarcando motores de videojuegos, software de diseño visual, composición de audio y bibliotecas de recursos (assets) libres de regalías, todas ellas bajo licencias del Instituto Tecnológico de Massachusetts (por sus siglas en inglés, MIT) y/o Licencia Pública General de GNU (por sus siglas en inglés, GNU GPL) que garantizan su uso sin restricciones comerciales independientemente del alcance del proyecto. A diferencia de soluciones propietarias como Unity o Unreal Engine, cuyo modelo de licenciamiento puede representar una barrera económica ante el éxito comercial, las herramientas aquí presentadas reducen dicha barrera, haciendo del tiempo y esfuerzo del desarrollador uno de los principales recursos requeridos. Los trabajos relacionados revisados respaldan la viabilidad del desarrollo independiente de videojuegos sin inversión económica, y evidencian que la ingeniería de software para videojuegos es un campo en crecimiento y maduración. Se concluye que existe un ecosistema completo y funcional de herramientas gratuitas que permite a desarrolladores principiantes llevar a cabo proyectos de videojuegos de principio a fin.

Palabras Clave:

Desarrollo de videojuegos, herramientas de código abierto, licencias de software libre, desarrollo independiente, motores de videojuegos.

Abstract:

Video game development has undergone a transformation in recent years. This process has been favored by the availability of free and open-source tools that cover all aspects of the creative process. This article analyzes accessible tools for beginner developers, encompassing game engines, visual design software, audio composition, and royalty-free resource (assets) libraries, all under Massachusetts Institute of Technology (MIT) and/or GNU General Public License (GNU GPL) licenses that guarantee unrestricted commercial use regardless of the project's scope. Unlike proprietary solutions such as Unity or Unreal Engine, whose licensing models may impose economic barriers upon commercial success, the tools presented here reduce such barriers, making time and developer effort the main resources required. Related works reviewed support the feasibility of independent video game development without financial investment, and demonstrate that software engineering for video games is a growing and maturing field. It is concluded that a complete and functional ecosystem of free tools exists, enabling beginner developers to carry out video game projects from start to finish.

Keywords:

Video game development, open-source tools, free software licenses, indie development, game engines.

IJISTA, Vol. II, Núm. I, Junio 2026,
ISSN 3122-4474

1. Introducción

La industria de los videojuegos ha experimentado un crecimiento dentro de la economía creativa global, con una tasa de crecimiento sostenida que atrae tanto a grandes corporaciones como a desarrolladores independientes [1]. Esta expansión no ha sido homogénea: mientras que los grandes estudios continúan apostando por producciones de alto presupuesto conocidas como títulos AAA (triple A), el segmento independiente ha experimentado un auge particularmente notable, impulsado por la proliferación de plataformas de distribución digital como Steam, itch.io y las tiendas de aplicaciones móviles [1].

Historicamente, el desarrollo de videojuegos ha sido dominado por grandes estudios con recursos para sustentar una producción prolongada, un equipo numeroso y múltiples capas de burocracia corporativa. Este modelo, imponía barreras de entrada significativas para desarrolladores sin respaldo institucional. Sin embargo, la evolución tecnológica y la madurez alcanzada por herramientas de desarrollo accesibles han transformado este panorama, haciendo posible que equipos pequeños o desarrolladores individuales produzcan y distribuyan videojuegos a escala global [2].

En este contexto, el desarrollo independiente de videojuegos ha ganado una relevancia creciente, tanto desde el punto de vista cultural como económico, un fenómeno que se observa en la aceptación de los videojuegos como tópico de periodismo especializado [3]. Títulos como Stardew Valley, Hollow Knight, Undertale y, más recientemente, Balatro desarrollado íntegramente con Love2D han demostrado que la calidad y el éxito comercial no son exclusivos de los grandes estudios. No obstante, uno de los desafíos persistentes para los desarrolladores independientes es la selección de herramientas adecuadas a sus capacidades técnicas, artísticas y, especialmente, financieras. En este sentido, el modelo de licenciamiento del software de desarrollo se convierte en un factor crítico, pues puede representar un riesgo económico significativo conforme el proyecto escala en popularidad.

A diferencia de soluciones comerciales como Unity o Unreal Engine, cuyo modelo de licenciamiento puede generar costos inesperados o modificaciones unilaterales de términos una vez que un proyecto alcanza determinado nivel de éxito, existe en la actualidad un ecosistema de herramientas de código abierto o de uso gratuito que garantizan condiciones estables independientemente del alcance comercial del proyecto [4]. Parte de estas herramientas opera bajo la licencia MIT, que permite su uso, redistribución y modificación sin restricciones; otra parte se distribuye bajo licencias GNU, que garantizan su uso completamente gratuito sin limitaciones comerciales.s A diferencia de alternativas propietarias, estas herramientas garantizan que los términos no cambiarán independientemente del alcance del proyecto.

Estas herramientas de acceso libre cubren gran parte del flujo de trabajo en el desarrollo de videojuegos. Para el desarrollo del juego sin necesidad de construir un motor desde cero, se emplean herramientas como Godot, un motor con licencia MIT que ha experimentado un crecimiento significativo en su adopción durante los últimos años y que soporta de forma nativa C++, C# y GDScript, su propio lenguaje de scripting inspirado en Python; así como Love2D, un framework basado en Lua que ha probado su viabilidad en proyectos comerciales de alto impacto [4]. En el ámbito visual, la elección de herramientas depende del estilo artístico que se busca lograr: Krita para ilustración 2D, Pixelorama para arte en pixel art, Blockbench para modelado 3D low poly, y Blender para producciones de mayor complejidad geométrica [5]. En el ámbito del audio, los desarrolladores utilizan herramientas como Audacity para la edición y producción de efectos de sonido, y en FamiStudio o Bosca Ceoil Blue para la composición de música original. Adicionalmente, existen repositorios de assets de uso libre como los Sonniss GDC Bundles, que ponen a disposición de la comunidad efectos de sonido libre de regalías (royalty-free) para su integración en proyectos comerciales.

A pesar de la relevancia práctica de este ecosistema de herramientas, existe una carencia notable de estudios académicos que lo documenten y analicen de manera sistemática. La mayoría de los recursos disponibles se concentran en tutoriales en línea o en foros de comunidades de práctica, pero pocas investigaciones formales han abordado la caracterización y comparación de estas alternativas en el contexto de desarrollo independiente. Este trabajo busca contribuir a subsanar esa brecha, ofreciendo un panorama estructurado del ecosistema de herramientas libres y gratuitas disponibles para el desarrollo independiente de videojuegos, con atención a sus modelos de licenciamiento, capacidades técnicas y casos de uso representativos.

El resto del artículo se organiza de la siguiente manera: la sección 2 presenta una revisión de la literatura relevante; la sección 3 describe la metodología empleada; la sección 4 presenta los resultados; la sección 5 expone el análisis de resultados; y la sección 6 presenta las conclusiones.

2. Trabajos Relacionados

Los trabajos relacionados se clasifican en cuatro categorías para su análisis comparativo que estructuran el estado del arte sobre el ecosistema de herramientas para el desarrollo independiente de videojuegos: (1) la industria y cultura del desarrollo independiente, (2) la ingeniería de software aplicada a videojuegos, (3) las

IJISTA, Vol. II, Núm. I, Junio 2026,
ISSN 3122-4474

herramientas libres y de código abierto, y (4) las herramientas en contextos educativos y de accesibilidad.

2.1. Industria y Cultura del Desarrollo Independiente de Videojuegos

La consolidación del desarrollador indie como figura relevante en la industria ha motivado un creciente cuerpo de investigación. Garda y Grabarczyk [1] argumentan que la independencia en el desarrollo de videojuegos no constituye una condición binaria, sino un espectro que involucra dimensiones económicas, creativas y distributivas, lo que implica que la elección de herramientas de desarrollo tiene implicaciones que van más allá de lo meramente técnico. En línea con esto, Garda y Grabarczyk [1] documentan la precariedad estructural que caracteriza al sector indie, señalando que estos desarrolladores operan en condiciones de incertidumbre financiera que los expone particularmente a variaciones en los costos operativos, incluidos los modelos de licenciamiento de herramientas.

Desde una perspectiva cultural e histórica, Sotamaa [2] analiza las culturas de desarrollo de videojuegos en el contexto finlandés mediante un enfoque de tres capas: regional, laboral y de estudio que permite contextualizar cómo factores históricos, regulatorios y organizacionales moldean las prácticas cotidianas de producción. El autor señala que el modelo clásico de publicación dependía fuertemente de herramientas propietarias, lo que restringía el acceso de estudios pequeños a los canales controlados por los grandes actores de la industria. De forma complementaria, Berg Marklund [6] et al. realizan una revisión de investigación empírica sobre desarrollo de videojuegos, identificando que los estudios académicos disponibles aún no cubren de manera exhaustiva las dimensiones prácticas del proceso productivo en contextos de pequeña escala. Por su parte, Moradzadeh [7] et al. presentan una revisión sistemática de la literatura sobre videojuegos en el ámbito del trabajo cooperativo asistido por computadora (CSCW), evidenciando que los videojuegos constituyen un dominio de estudio multidimensional cuyas dinámicas de colaboración y producción merecen atención académica continua.

En cuanto a los aspectos legales y contractuales del sector, Gonçalves y Paim [8] analizan la integración de licencias mecánicas para videojuegos en contextos normativos de distintos países, destacando la complejidad que enfrentan los desarrolladores independientes al gestionar derechos de uso de contenido, lo que refuerza la importancia de contar con herramientas con modelos de licenciamiento claros y accesibles.

2.2. Ingeniería de Software para Videojuegos y Motores de Juego

El desarrollo de videojuegos ha madurado como disciplina técnica. Actualmente constituye un campo con características propias dentro de la ingeniería de software. Chueca et al. [9] presentan una revisión sistemática de la literatura sobre Ingeniería de Software para Videojuegos (GSE, por sus siglas en inglés) a escala industrial, confirmando que se trata de un dominio independiente y en crecimiento, con una producción investigativa cuatro veces mayor que antes de 2009, y con temas emergentes como arquitectura, diseño y gestión de calidad. Scacchi [10] ofrece una panorámica de las prácticas y tecnologías en la ingeniería de software para videojuegos, describiendo una amplia variedad de kits de desarrollo, motores y marcos de trabajo tanto comerciales como de código abierto orientados a distintos géneros y plataformas. El autor identifica que los motores de videojuegos representan uno de los casos de éxito más representativos en materia de reutilización de software, aunque también advierte que algunos estudios evitan adoptar motores comerciales disponibles por temor a que sus características limiten la originalidad del producto final.

Desde una perspectiva comparativa, Politowski [11] et al. exploran si los motores de videojuegos pueden considerarse marcos de software (frameworks) a partir de tres perspectivas: literatura, de código y humana. Analizando 282 motores de código abierto en GitHub, concluyen que existen diferencias cualitativas pero no cuantitativas significativas respecto a los frameworks tradicionales, siendo los motores generalmente más grandes, complejos y con comunidades más pequeñas. En una línea complementaria, Ullmann [12] et al. realizan un análisis anatómico comparativo de motores de videojuegos de código abierto, identificando patrones arquitectónicos comunes y diferencias estructurales entre distintas categorías de motores. Toftedahl [4] y Engström, por su parte, proponen una taxonomía de motores de videojuegos dividida en cuatro tipos motor central, motor de juego, motor de propósito general y motor de propósito especial y analizan su adopción en plataformas de distribución como Steam e Itch.io, encontrando que Unity concentra el 47.3% de los juegos independientes en esta última plataforma.

Sobota y Pietriková [13] abordan el rol de los motores de videojuegos tanto en el proceso de desarrollo como en la enseñanza de ciencias de la computación, destacando que su uso en entornos educativos facilita la comprensión de conceptos complejos de ingeniería de software. En el ámbito de las pruebas de software, Politowski, Petrillo y Guéhéneuc [14] realizan un estudio exhaustivo sobre las prácticas de prueba en videojuegos, identificando que este es uno de los procesos menos estandarizados en la industria y que la mayoría de las herramientas disponibles aún no están integradas de

IJISTA, Vol. II, Núm. I, Junio 2026,
ISSN 3122-4474

forma nativa en los principales motores. Asimismo, Kamienski y Bezemer [15] realizan un estudio empírico sobre el uso de sitios de preguntas y respuestas como Stack Overflow por parte de desarrolladores de videojuegos, encontrando que estas comunidades constituyen un recurso fundamental de soporte técnico para quienes trabajan con motores tanto comerciales como de código abierto.

2.3. Herramientas Libres, de Código Abierto y Contenido Procedural

La investigación específica sobre herramientas libres para el desarrollo de videojuegos es aún escasa, pero creciente. Hall, Dogbe y Said en [16]. presentan una revisión sistemática de la literatura sobre desarrollo de videojuegos de código abierto, identificando las principales herramientas disponibles, sus casos de uso y los factores que inciden en su adopción por parte de desarrolladores independientes. Los autores señalan que, a pesar del crecimiento sostenido del ecosistema de herramientas libres, la literatura académica sobre sus características, limitaciones y modelos de gobernanza sigue siendo insuficiente.

En el ámbito de la generación de contenido, Zamorano, Cetina y Sarro [17] presentan una revisión exhaustiva de técnicas de generación procedural de contenido asistida por búsqueda (SBPCG) aplicadas a videojuegos, clasificando las herramientas y enfoques disponibles según el tipo de contenido generado. Esta área resulta relevante para el ecosistema de herramientas libres, dado que muchos de estos métodos se implementan como bibliotecas de código abierto. Sweetser [18] realiza un análisis preliminar del estado de la investigación sobre la integración de modelos de lenguaje de gran escala (LLMs) en videojuegos, abarcando desde la generación de contenido narrativo hasta la asistencia en el proceso de desarrollo mismo, lo cual abre nuevas posibilidades para herramientas accesibles basadas en inteligencia artificial. Hu [19] et al. complementan este panorama con una revisión sobre el uso de videojuegos como entornos de entrenamiento y evaluación para sistemas de inteligencia artificial, destacando la importancia de plataformas abiertas y reproducibles para la investigación en este campo.

2.4. Herramientas en Contextos Educativos y Accesibilidad

La literatura sobre tecnología educativa aporta evidencia relevante acerca del impacto que tiene la disponibilidad de herramientas accesibles en la participación y producción creativa. Christopoulos, Conrad y Shukla [5] analizan cómo entornos virtuales accesibles y de bajo costo inciden directamente en los niveles de participación y producción creativa de los estudiantes, un hallazgo extrapolable al contexto del desarrollo independiente de videojuegos.

De manera más específica, Sáez-López et al. [20] evalúan una propuesta pedagógica para la formación inicial de docentes que combina el entorno de programación visual por bloques Scratch con el motor Unity, utilizando actividades progresivas en C# como puente entre ambos entornos. Los resultados muestran mejoras estadísticamente significativas tanto en el rendimiento como en las actitudes hacia la programación, lo que refuerza la viabilidad de introducir herramientas de desarrollo de videojuegos incluyendo el uso de motores gratuitos en contextos educativos formales. Este estudio resulta especialmente relevante para el presente trabajo, ya que Unity constituye uno de los motores gratuitos de mayor adopción en el contexto indie y su uso en educación ilustra el potencial democratizador de las herramientas de acceso libre.

De la revisión presentada se desprende que, si bien existe un conjunto amplio de investigaciones sobre motores comerciales, culturas de desarrollo y prácticas de ingeniería de software para videojuegos, la caracterización sistemática del ecosistema de herramientas libres y gratuitas con atención a sus modelos de licenciamiento, capacidades técnicas por área de producción y pertinencia para el contexto indie actual constituye un vacío que este estudio busca contribuye a cerrar.

3. Metodología

El presente trabajo adopta un enfoque de revisión estructurada y análisis comparativo, orientado a caracterizar el ecosistema de herramientas libres y gratuitas disponibles para el desarrollo independiente de videojuegos. La metodología se organiza en tres etapas: definición de criterios de selección, recopilación y clasificación de herramientas, y análisis comparativo por área de producción.

3.1. Criterios de Selección

Para delimitar el conjunto de herramientas a analizar se establecieron los siguientes criterios de inclusión:

- La herramienta debe ser de uso gratuito, ya sea bajo una licencia de código abierto (MIT, GNU GPL, Apache) o mediante un modelo freeware sin restricciones comerciales.
- Debe cubrir al menos una de las áreas del flujo de trabajo en el desarrollo de videojuegos: motor de juego, arte 2D, arte 3D, audio o gestión de assets.
- Debe contar con documentación oficial activa y una comunidad de usuarios verificable, a través de repositorios públicos o foros oficiales.

IJISTA, Vol. II, Núm. I, Junio 2026,
ISSN 3122-4474

- Debe haber sido utilizada en al menos un proyecto de distribución pública, en plataformas como Steam, itch.io, repositorios públicos o el sitio web oficial del proyecto, ya sea comercial o no comercial.
- Se excluyeron herramientas con modelos freemium que imponen restricciones comerciales al superar ciertos umbrales de ingresos, así como aquellas cuya documentación oficial se encuentra descontinuada o sin mantenimiento activo.

3.2. Recopilación y Clasificación de Herramientas

La recopilación de herramientas se realizó a partir de tres fuentes principales: literatura académica revisada en la sección 2, documentación oficial de plataformas de distribución de videojuegos independientes (Steam e itch.io), y repositorios públicos en GitHub. Las herramientas identificadas fueron clasificadas según el área de producción a la que corresponden, dando lugar a cinco categorías, como se muestra en la Tabla 1.

Tabla 1. Categorías de herramientas libres para el desarrollo independiente de videojuegos.

Fuente: elaboración propia.

Categoría	Descripción	Ejemplos Identificados
Motor de juego	Entorno principal de desarrollo y ejecución	Godot, Love2D
Arte 2D	Ilustración, animación y pixel art	Krita, Pixelorama
Arte 3D	Modelado y animación tridimensional	Blender, Blockbench
Audio	Edición de sonido y composición musical	Audacity, FamiStudio, Bosca Ceoil Blue
Assets libres	Repositorios de recursos reutilizables	Sonniss GDC Bundles

3.3. Análisis Comparativo

Se analizó cada herramienta con base en cuatro dimensiones: (1) modelo de licenciamiento y sus implicaciones comerciales, (2) capacidades técnicas principales, (3) plataformas de destino soportadas, y (4) casos de uso representativos en proyectos publicados. Este análisis se sintetiza en la sección 4 mediante tablas comparativas y se complementa con una discusión sobre las fortalezas y limitaciones de cada herramienta en el contexto del desarrollo indie. El flujo general de la metodología se ilustra en la Fig. 1.

Figura 1. Flujo metodológico del estudio. Elaboración propia

3.4. Evaluación de Modelos de Licenciamiento

Dado que uno de los ejes centrales de este trabajo es el impacto del licenciamiento en la viabilidad económica del desarrollo independiente, se realizó un análisis específico de los términos legales de cada herramienta identificada, evaluando cuatro aspectos: restricciones de uso comercial, condiciones de redistribución, restricciones sobre la modificación del código fuente y los términos ante el éxito comercial del proyecto. Para ello se consultaron directamente los textos de licencia disponibles en los repositorios oficiales y en la documentación publicada por cada proyecto. Las licencias fueron clasificadas en tres categorías según el grado de libertad que otorgan sobre el uso comercial y la redistribución de proyectos y código derivado.

Las licencias permisivas, como MIT y Apache 2.0, permiten el uso, modificación y distribución comercial sin restricciones ni obligación de publicar el código fuente del producto derivado. Este tipo de licencia representa la opción de menor riesgo contractual para el desarrollador independiente, ya que garantiza estabilidad en los términos independientemente del éxito comercial del proyecto.

Las licencias copyleft, como GNU GPL y GNU LGPL, garantizan el uso completamente gratuito sin limitaciones comerciales, pero imponen condiciones sobre la redistribución del código. En la práctica, esto significa que, si el desarrollador modifica la herramienta y distribuye el resultado, está obligado a publicar esas modificaciones bajo la misma licencia.

Los modelos freeware sin código abierto corresponden a herramientas gratuitas cuyo código fuente no está disponible públicamente. Si bien no implican un costo directo, generan una dependencia total del proveedor para actualizaciones, correcciones y continuidad del proyecto, lo que representa un riesgo a largo plazo comparable al de las soluciones comerciales.

IJISTA, Vol. II, Núm. I, Junio 2026,
ISSN 3122-4474

Este análisis permite al desarrollador independiente no solo identificar qué herramientas son gratuitas en el momento de adoptarlas, sino también anticipar qué condiciones seguirán vigentes conforme el proyecto escale en popularidad o ingresos, reduciendo el riesgo de enfrentar modificaciones unilaterales de términos como las documentadas en herramientas comerciales de amplia adopción [4].

4. Resultados

A continuación, se presentan los datos obtenidos del análisis comparativo de las herramientas libres identificadas, organizados por categoría de producción. Para cada categoría se sintetizan el modelo de licenciamiento, las capacidades técnicas principales, las plataformas de destino soportadas y los casos de uso representativos en proyectos publicados.

Tabla 2. Comparativa de motores de videojuego libres para desarrollo independiente.

Fuente: elaboración propia.

Herramienta	Licencia	Lenguajes	Plataformas destino	Caso de uso representativo
Godot 4	MIT	GDScript, C#, C++	Windows, macOS, Linux, Android, iOS, Web	Usado en títulos publicados en Steam e itch.io; base de 1,5 M de usuarios activos según la propia comunidad
Love2D	MIT	Lua	Windows, macOS, Linux, Android, iOS	LuaBalatro (2024), uno de los juegos indie más vendidos de su año, desarrollado íntegramente con este framework

Tabla 3. Comparativa de herramientas libres para arte 2D.
Fuente: elaboración propia.

Herramienta	Licencia	Especialidad	Formatos de exportación	Caso de uso representativo
Krita	GNU GPL v3	Ilustración y pintura digital	PNG, PSD, SVG, TIFF, WebP	Creación de concept art, sprites y fondos para proyectos 2D; adoptada por ilustradores en títulos indie publicados en plataformas digitales
Pixelorama	MIT	Pixel art y animación frame-by-frame	PNG, GIF, APNG, spritesheet	Diseño de sprites animados para juegos de plataformas y RPGs; integración directa con Godot mediante el formato de proyecto compartido

Tabla 4. Comparativa de herramientas libres para arte 3D.

Fuente: elaboración propia.

Herramienta	Licencia	Especialidad	Formatos de exportación	Caso de uso representativo
Blender	GNU GPL v2+	Modelado, rigging, animación y render	FBX, OBJ, GLTF/GLB, DAE, USD	Producción de personajes, escenarios y cinemáticas; adoptado en producciones indie de complejidad media-alta con exportación directa a Godot y Unity
Blockbench	MIT	Modelado low poly y voxel	OBJ, FBX, GLTF, JSON (Bedrock)	Diseño de modelos para juegos de estilo minimalista y mods de Minecraft; flujo de trabajo orientado a principiantes con interfaz simplificada

Tabla 5. Comparativa de herramientas libres para audio.

Fuente: elaboración propia.

Herramienta	Licencia	Especialidad	Formatos de exportación	Caso de uso representativo
Audacity	GNU GPL v2	Edición y producción de audio	WAV, MP3, OGG, FLAC, AIFF	Grabación, limpieza y procesamiento de efectos de sonido para integración en motores de videojuegos
FamiStudio	MIT	Composición de música chiptune (NES)	WAV, OGG, NSF, ROM NES	Composición de bandas sonoras con estética retro para juegos de plataformas y RPGs de 8 bits
Bosca Ceoil Blue	MIT	Composición de música electrónica y chiptune	WAV, OGG, MID	Creación de música original para proyectos de jam de videojuegos; curva de aprendizaje reducida orientada a desarrolladores sin formación musical

Tabla 6. Repositorios de assets libres de regalías identificados.
Fuente: elaboración propia.

Recurso	Licencia	Tipo de contenido	Uso comercial	Caso de uso representativo
Sonniss GDC Bundles	Royalty-free (custom)	Efectos de sonido	Sí, sin restricciones	Bibliotecas de efectos de sonido distribuidas anualmente durante la Game Developers Conference; utilizadas en proyectos indie publicados comercialmente

IJISTA, Vol. II, Núm. I, Junio 2026,
ISSN 3122-4474

Tabla 7. Distribución de herramientas por tipo de licencia.

Fuente: elaboración propia.

Tipo de licencia	Herramientas incluidas	Permite uso comercial sin restricciones	Requiere publicar modificaciones
Permisiva (MIT)	Godot, Love2D, Pixelorama, Blockbench, FamiStudio, Bosca Ceoil Blue	Sí	No
Copyleft (GNU GPL)	Krita, Blender, Audacity	Sí	Solo si se redistribuye la herramienta modificada
Royalty-free (custom)	Sonniss GDC Bundles	Sí	No aplica

5. Análisis de Resultados

Los resultados presentados en la sección anterior sugieren la existencia de un ecosistema completo de herramientas libres que cubre íntegramente el flujo de trabajo en el desarrollo independiente de videojuegos, desde la programación del motor hasta la producción de audio, sin requerir inversión económica significativa en licencias. A continuación, se interpreta el significado de estos hallazgos en relación con el contexto planteado en la introducción.

En primer lugar, el análisis del modelo de licenciamiento (Tabla 7) revela que la mayoría de las herramientas identificadas operan bajo licencias MIT, consideradas las más permisivas del espectro del software libre. Esta distribución es especialmente relevante en el contexto indie: a diferencia de licencias copyleft como GNU GPL, las licencias MIT no imponen condiciones sobre el código propietario del videojuego resultante, reduciendo la incertidumbre legal al escalar un proyecto hacia la distribución comercial. Este hallazgo es consistente con el riesgo documentado en motores propietarios como Unity, cuyas modificaciones unilaterales de términos en 2023 generaron una crisis de confianza en el sector independiente [4], y refuerza la pertinencia de las alternativas aquí analizadas.

En cuanto a los motores de videojuego (Tabla 2), los datos sugieren que Godot y Love2D representan opciones viables para proyectos de distinto perfil, considerando el soporte multiplataforma, disponibilidad de documentación y una comunidad activa. Godot destaca por su arquitectura basada en nodos, su soporte nativo para exportación multiplataforma y su crecimiento sostenido en adopción desde 2020, impulsado en parte por el éxodo de desarrolladores que abandonaron Unity tras las controversias de licenciamiento mencionadas. Love2D, por su parte, ha demostrado viabilidad comercial a través de títulos como Balatro o Move or Die, casos que sugieren que un framework de menor escala puede sustentar productos comerciales exitosos, aunque no constituyen evidencia generalizable. En conjunto, ambas herramientas cubren un amplio espectro de necesidades de un desarrollador principiante: Godot ofrece un entorno visual integrado más accesible, mientras que Love2D favorece un enfoque más programático orientado a quienes ya dominan la lógica de código.

Respecto a las herramientas visuales (Tablas 3 y 4), se observa una complementariedad natural entre las opciones identificadas. En el ámbito 2D, Krita y Pixelorama atienden estilos artísticos diferenciados: ilustración de alta fidelidad y pixel art, respectivamente, lo que permite al desarrollador seleccionar la herramienta según la estética del proyecto. En el ámbito 3D, la distinción entre Blender y Blockbench refleja una jerarquía de complejidad técnica: Blender es la herramienta de referencia para producciones de mayor exigencia geométrica y animación, mientras que Blockbench reduce la barrera de entrada para quienes se aproximan al modelado 3D por primera vez. Esta distribución es coherente con el enfoque del trabajo, centrado en desarrolladores principiantes que pueden progresar dentro del mismo ecosistema libre conforme crecen sus capacidades técnicas.

En el área de audio (Tabla 5), los resultados muestran que las tres herramientas identificadas cubren las dos dimensiones principales de la producción sonora para videojuegos: edición técnica de audio (Audacity) y composición musical original (FamiStudio, Bosca Ceoil Blue). La especialización chiptune de FamiStudio y Bosca Ceoil Blue no debe interpretarse como una limitación, sino como una ventaja contextual para el perfil de desarrollador principiante: la música de 8 bits reduce la complejidad compositiva al trabajar con síntesis paramétrica acotada, lo que facilita la producción de bandas sonoras originales sin formación musical previa. Audacity, por su parte, es suficiente para el procesamiento de cualquier efecto de sonido en proyectos de escala indie, cubriendo tareas desde la eliminación de ruido hasta la mezcla multicanal básica.

Finalmente, la inclusión de los Sonniss GDC Bundles como repositorio de assets libres (Tabla 6) aborda una dimensión práctica frecuentemente ignorada en la literatura: la disponibilidad de recursos de alta calidad que pueden integrarse directamente sin necesidad de producirlos desde cero. Esto es especialmente relevante en las etapas tempranas del desarrollo, cuando el desarrollador principiante puede priorizar la mecánica del juego sobre la producción de todos sus activos. En conjunto, los resultados respaldan la hipótesis central de este trabajo: existe un ecosistema funcional y maduro de herramientas libres que, tomadas en su conjunto, proporcionan al desarrollador independiente todo lo necesario para llevar un proyecto de videojuego de la concepción a la

distribución comercial sin incurrir en costos de licenciamiento.

6. Conclusiones

El presente trabajo ha caracterizado un ecosistema de nueve herramientas libres y gratuitas distribuidas en cinco categorías de producción (motores de videojuego, arte 2D, arte 3D, audio y assets), todas ellas bajo licencias que favorecen el uso comercial sin restricciones ni riesgo de modificaciones unilaterales de términos. El análisis realizado permite formular las siguientes conclusiones.

Primero, el ecosistema de herramientas libres para el desarrollo de videojuegos ha alcanzado un nivel de madurez suficiente para sustentar proyectos comerciales de impacto, como lo demuestra los casos de Balatro o Move or Die con Love2D, y Buckshot Roulette, Brotato o Webfishing con Godot. Esto representa un cambio estructural respecto al paradigma descrito en la introducción, en el que los grandes estudios concentraban el acceso a las herramientas de producción profesional.

Segundo, la predominancia de licencias MIT en el ecosistema analizado constituye una ventaja diferencial frente a las alternativas propietarias: garantiza estabilidad contractual independientemente del éxito comercial del proyecto, reduciendo el riesgo documentado en motores como Unity. Esta estabilidad es un factor crítico para el desarrollador indie, cuya precariedad financiera lo hace especialmente vulnerable a cambios en los términos de licenciamiento [1].

Tercero, el conjunto de herramientas identificado cubre gran parte del flujo de trabajo en el desarrollo de videojuegos, desde el motor y el diseño visual hasta la producción de audio y la gestión de assets, lo que convierte al tiempo y esfuerzo del desarrollador en los principales recursos requeridos para ejecutar un proyecto de principio a fin.

Como trabajo futuro, se proponen tres líneas de investigación. En primer lugar, la realización de estudios empíricos comparativos que midan el impacto del uso de herramientas libres frente a soluciones propietarias sobre la productividad, la calidad del producto final y la viabilidad económica de proyectos indie reales. En segundo lugar, la evaluación de la integración de inteligencia artificial generativa, particularmente modelos de lenguaje de gran escala, en el flujo de trabajo con herramientas libres, dado el creciente interés académico en este campo [18]. En tercer lugar, la extensión de este análisis a contextos educativos formales, explorando el potencial de este ecosistema para democratizar la enseñanza del desarrollo de videojuegos en instituciones con recursos tecnológicos limitados, en línea con los hallazgos de Christopoulos et al. [5] y Sáez-López et al. [20] sobre el impacto positivo de herramientas accesibles en la participación y producción creativa de los estudiantes.

7. Referencias

[1] Garda, M. B., & Grabarczyk, P. (2016). Is every indie game independent? Towards the concept of indie game. Game Studies, 16(1). http://gamestudies.org/1601/articles/gardagrabarczyk

[2] Sotamaa, O. (2021). Studying game development cultures. Games and Culture. https://doi.org/10.1177/15554120211005242

[3] Foxman, Maxwell & Nieborg, David. (2016). Between a Rock and a Hard Place: Games Coverage and its Network of Ambivalences. Journal of Games Criticism. 3. https://www.researchgate.net/publication/296484250_Between_a_Rock_and_a_Hard_Place_Games_Coverage_and_its_Network_of_Ambivalences

[4] Toftedahl, M., & Engström, H. (2019). A taxonomy of game engines and the tools that drive the industry. DiGRA Conference Proceedings. https://doi.org/10.26503/dl.v2019i1.1077

[5] Christopoulos, A., Conrad, M., & Shukla, M. (2018). Increasing student engagement through virtual and gamified environments. Virtual Reality 22(4). https://doi.org/10.1007/s10055-017-0330-3

[6] Berg Marklund, B., Engström, H., Hellkvist, M., & Backlund, P. (2019). What empirically based research tells us about game development. The Computer Games Journal, 8, 179–198. https://doi.org/10.1007/s40869-019-00085-1

[7] Moradzadeh, S., Zhang, Z., Gui, X., & Kou, Y. (2025). The state of video game research in computer-supported cooperative work: A systematic literature review. Computer Supported Cooperative Work, 34, 587–638. https://doi.org/10.1007/s10606-025-09519-z

[8] Gonçalves, L. R., & Paim, B. W. (2023). How to integrate mechanical licences for video games in the United States, the United Kingdom and China. Journal of Gaming & Virtual Worlds, 15, 161–175. https://doi.org/10.1386/jgvw_00078_

[9] Chueca, J., Verón, J., Font, J., Pérez, F., & Cetina, C. (2024). The consolidation of game software engineering: A systematic literature review of software engineering for industry-scale computer games. Information and Software Technology, 165. https://doi.org/10.1016/j.infsof.2023.107330

[10] Scacchi, W. (2017). Practices and technologies in computer game software engineering. IEEE Software, 34(1), 110–116. https://doi.org/10.1109/MS.2017.20

[11] Politowski, C., Petrillo, F., Montandon, J. E., Valente, M. T., & Guéhéneuc, Y.-G. (2020). Are game engines software frameworks? A three-perspective study. Journal of Systems and Software, 171. https://doi.org/10.1016/j.jss.2020.110846

[12] Ullmann, G. C., Politowski, C., Guéhéneuc, Y.-G., & Petrillo, F. (2022). Game engine comparative anatomy. arXiv:2207.06473 [cs.SE]. https://doi.org/10.48550/arXiv.2207.06473

[13] Sobota, B., & Pietriková, E. (2023). The role of game engines in game development and teaching. In Computer Science for Game Development and Game Development for Computer Science. IntechOpen. https://doi.org/10.5772/intechopen.1002257

IJISTA, Vol. II, Núm. I, Junio 2026,
ISSN 3122-4474

[14] Politowski, C., Petrillo, F., & Guéhéneuc, Y.-G. (2021). A survey of video game testing. arXiv:2103.06431 [cs.SE]. https://doi.org/10.48550/arXiv.2103.06431

[15] Kamienski, A., & Bezemer, C.-P. (2021). An empirical study of Q&A websites for game developers. Empirical Software Engineering, 26, 115. https://doi.org/10.1007/s10664-021-10014-4

[16] Hall, H., Dogbe, A., & Said, H. (2025). Open-source game development: A systematic literature review. In M. Jones (Ed.), Proceedings of InSITE 2025: Informing Science and Information Technology Education Conference, Article 20. Informing Science Institute. https://doi.org/10.28945/5563

[17] Zamorano, M., Cetina, C., & Sarro, F. (2023). The quest for content: A survey of search-based procedural content generation for video games. arXiv:2311.04710 [cs.SE]. https://doi.org/10.48550/arXiv.2311.04710

[18] Sweetser, P. (2024). Large language models and video games: A preliminary scoping review. arXiv:2403.02613 [cs.HC]. https://doi.org/10.48550/arXiv.2403.02613

[19] Hu, C., Zhao, Y., Wang, Z., Du, H., & Liu, J. (2024). Games for artificial intelligence research: A review and perspectives. arXiv:2304.13269 [cs.AI]. https://doi.org/10.48550/arXiv.2304.13269

[20] Sáez-López, J. M., González-Calero, J. A., Cózar-Gutierrez, R., & del Olmo-Muñoz, J. (2023). Scratch and unity design in elementary education: A study in initial teacher training. Journal of Computer Assisted Learning, 39(5), 1528–1538. https://doi.org/10.1111/jcal.12815

Próximo Número Diciembre 2026 – Mayo 2027

Convocatoria abierta para recepción de artículos

Directrices para autores disponible en:

https://revistas.uas.edu.mx/index.php/IJISTA/envios

Hecho en México.
Sitio web administrado por:
Cuerpo Académico Realidad Virtual y Robótica (UAS-254)
Facultad de Informática Culiacán de la Universidad Autónoma de Sinaloa
Correo electrónico: editor.ijista@uas.edu.mx
International Journal of Information Science and Technological Applications-UAS IJISTA
Disponible en: https://revistas.uas.edu.mx/index.php/IJISTA

International Journal of Information Science and Technological Applications-UAS

IJISTA

Vol. II, Núm. II, Junio 2026

ISSN: 3122-4474