ISSN: en trámite

Fecha de recepción: 06 de junio de 2025, fecha de publicación en línea: octubre de 2025.

Análisis de Desempeño de Servicios de Notificación en la Nube: Twilio y Google SMTP

Blanca Estela Islas Flores[1] , Arley Iván Solis Zacapantzi[2] ,María Janaí Sánchez Hernández[3] , José Juan Hernández Mora[4], Juan Ramos Ramos[5], Elizabeth Cuatecontzi Cuahutle. [6]

[1] Tecnológico Nacional de México – Instituto Tecnológico de Apizaco. San Andrés Ahuashuatepec, Municipio de Tzompantepec, Tlaxcala, C.P. 90491, México.

Autor de correspondencia: Autor. Blanca Estela Islas Flores (m23370004@apizaco.tecnm.mx).

Abstract- This study provides a practical foundation for optimizing notification systems in critical applications, combining efficiency, security, and scalability This study analyzes the performance of Twilio and Google SMTP (Simple Mail Transfer Protocol) cloud-based notification services, focusing on latency, costs, and CPU usage rather than proposing a new software design. Using the Extreme Programming (XP) methodology to structure experimental iterations—selected for its emphasis on rapid feedback and iterative improvement—we evaluated key metrics across various hardware platforms (Intel® Core™ i3, i5, i7 and Apple M1). Controlled tests reveal that Google SMTP achieves lower latency subject to throughput limitations, while Twilio delivers consistent messaging performance at a fixed per-message cost. Our results provide practical insights for optimizing real-time notification systems in critical applications. Future work will explore integration of more scalable technologies, such as Firebase Cloud Messaging and RabbitMQ.

Keywords: Extreme Programming (XP), Google SMTP, performance, real-time notifications, scalability, software architecture, Twilio.

I. INTRODUCCIÓN

Las notificaciones en tiempo real son elementos fundamentales en los sistemas modernos; sin embargo, su implementación conlleva desafíos significativos relacionados con la latencia, los costos operativos y el uso de recursos de cómputo.

En este estudio se analiza el rendimiento de dos servicios de notificación en la nube Twilio (mensajes SMS) y Google Simple Mail Transfer Protocol (SMTP) mediante un conjunto de pruebas controladas que comparan su eficiencia sobre distintos procesadores (Intel® Core™ i3, i5, i7 y Apple M1).

Se eligieron Twilio y Google SMTP porque representan dos canales de notificación complementarios SMS y correo electrónico con alta adopción industrial, API REST maduras y planes gratuitos comparables. Esta selección permite aislar el efecto del tipo de canal sobre el desempeño sin introducir sesgos derivados de proveedores con arquitecturas similares; alternativas como AWS SNS, SendGrid o Firebase Cloud Messaging se descartaron para mantener el experimento controlado y la carga de pruebas manejable.

El empleo de plataformas *cloud* ha demostrado reducir tanto los tiempos de desarrollo como los costos iniciales gracias a su rápida configuración, alta disponibilidad y escalabilidad elástica. Según Alfaro Quintero et al. [1], las herramientas gratuitas basadas en infraestructura en la nube permiten entrenar modelos complejos con barreras mínimas de entrada técnica y económica, optimizando procesos críticos como la detección temprana de enfermedades. Estas ventajas también resultan aplicables al diseño de sistemas de notificaciones, facilitando la implementación de arquitecturas eficientes sin grandes inversiones iniciales.

El resto del artículo se organiza de la siguiente manera: el Apartado II presenta los trabajos relacionados que abordan la implementación de notificaciones en tiempo real mediante APIs en la nube; el Apartado III describe el estado actual del problema y los principales retos técnicos; el Apartado IV detalla la metodología experimental basada en Extreme Programming (XP); el Apartado V expone y discute los resultados obtenidos; y, finalmente, el Apartado VI resume las conclusiones y propone futuras líneas de investigación, incluida la integración de Firebase Cloud Messaging y RabbitMQ.

II. TRABAJOS RELACIONADOS

Diversos estudios recientes han explorado la implementación de notificaciones en tiempo real mediante APIs y arquitecturas distribuidas en la nube, subrayando tanto los avances conseguidos como los desafíos por superar. Uchuari Quiñónez [2] desarrolló un sistema de gestión de citas y servicios para la peluquería "Shakinah" empleando el patrón Model-View-ViewModel (MVVM), demostrando la viabilidad de integrar notificaciones instantáneas en contextos de salud digital. Trujillo Malaver, Daza Rojas y Morales Gallego [3] propusieron una mejora en la arquitectura del sistema de gestión de la demanda WA Collaborative, basada en microservicios .NET y orquestada con RabbitMQ, alcanzando mejoras en escalabilidad y balanceo de carga. Prabhakar et al. [4] emplearon aprendizaje por refuerzo offline para optimizar múltiples objetivos en el envío de notificaciones móviles, evidenciando aumentos en eficiencia y personalización. Yuan et al. [5] exploraron políticas de aprendizaje por refuerzo offline para notificaciones móviles en entornos con restricciones de entrega, mostrando la viabilidad de estrategias adaptativas. Finalmente, Samarin et al. [6] investigaron cómo las aplicaciones de mensajería segura filtran datos sensibles a servicios push, revelando vulnerabilidades de privacidad y proponiendo mecanismos de mitigación. Estos aportes proporcionan un marco técnico sólido que respalda la comparación entre Twilio y Google SMTP, estableciendo criterios de evaluación centrados en rendimiento, seguridad y escalabilidad.

De los trabajos analizados, destacan particularmente las contribuciones de Trujillo Malaver et al. y Prabhakar et al., quienes, respectivamente, demostraron la eficacia de arquitecturas multicapa basadas en microservicios con RabbitMQ y de políticas de aprendizaje por refuerzo offline para optimizar la entrega de notificaciones. **Estos enfoques** fundamentan la elección de Twilio y Google SMTP y refuerzan el marco técnico de nuestro análisis comparativo.

III. ESTADO DEL PROBLEMA

Las notificaciones en tiempo real son fundamentales en diversas aplicaciones y servicios, desde la comunicación empresarial hasta la gestión de sistemas administrativos y de seguridad. Sin embargo, su implementación eficiente enfrenta varios desafíos técnicos y operativos.

Uno de los principales problemas es la latencia en la entrega de mensajes, que puede verse afectada por factores como la infraestructura del servidor, la velocidad de la red y la optimización del código de las APIs utilizadas. Un retraso en la entrega de notificaciones puede afectar la experiencia del usuario y la funcionalidad del sistema.

Otro aspecto crítico es el costo asociado al uso de servicios de notificación en la nube. Muchas soluciones dependen de plataformas externas como Twilio y Firebase Cloud Messaging, cuyos costos pueden escalar rápidamente en función del número de notificaciones enviadas. Esto plantea un desafío para organizaciones que buscan optimizar el gasto sin comprometer la eficiencia y confiabilidad del servicio.

Además, el uso de CPU y la carga sobre los servidores son elementos determinantes en la arquitectura de los sistemas de notificaciones. Un mal diseño en la implementación puede generar sobrecarga en los servidores, aumentando el consumo de recursos y afectando el rendimiento general del sistema. Es fundamental considerar estrategias como la distribución de carga, uso de colas de mensajes y optimización de consultas a bases de datos para mitigar estos efectos.

Finalmente, la seguridad en las notificaciones es un aspecto crítico, especialmente en contextos donde la información enviada contiene datos sensibles. La falta de cifrado adecuado o la exposición de credenciales de API pueden generar vulnerabilidades que comprometan la integridad y privacidad de los usuarios. La implementación de protocolos de seguridad robustos y el monitoreo constante del tráfico de notificaciones son esenciales para garantizar un sistema seguro y confiable.

IV. METODOLOGÍA

La presente investigación se desarrolló bajo la metodología Extreme Programming (XP), caracterizada por su enfoque iterativo e incremental, orientado a la mejora continua y la entrega temprana de resultado como se muestra en la **Figura 1.**

Figura 1. Diagrama XP

¿Por qué Extreme Programming (XP)?

XP fue seleccionado por sus ciclos de desarrollo muy cortos, integración continua y feedback constante del cliente. A diferencia de metodologías como Scrum o Waterfall, XP facilita la rápida adaptación a cambios de requisitos, promueve el desarrollo guiado por pruebas (Test-Driven Development, TDD) y fomenta la programación en parejas para garantizar la calidad del código.

El proceso metodológico se estructuró en cinco fases principales:

A. Planificación

En esta fase se definieron los objetivos del estudio y las métricas clave de evaluación: latencia, costo por notificación y uso de CPU. Asimismo, se seleccionaron las plataformas Twilio y Google SMTP (correo electrónico), y se establecieron los escenarios de carga con 10, 50, 100 y 500 notificaciones para su análisis.

B. Diseño

Se diseñó una plataforma de pruebas utilizando el lenguaje de programación Python y el framework Django, adoptando una arquitectura híbrida que integró microservicios, una base de datos relacional PostgreSQL y mecanismos para el registro de métricas. Este diseño permitió centralizar el monitoreo del sistema y facilitar la recolección de datos durante los experimentos.

C. Codificación

Siguiendo los principios de Test- Driven Development (TDD), se desarrollaron pruebas automatizadas mediante la herramienta pytest. Las pruebas fueron ejecutadas en distintos entornos de hardware —Intel Core i3, i5, i7 y Apple M1— con el objetivo de evaluar el impacto del procesador en el rendimiento del sistema.

D. Validación y análisis de resultados

Se realizaron múltiples iteraciones para evaluar comparativamente el desempeño de Twilio y Google *SMTP* en condiciones controladas. Además, se llevaron a cabo simulaciones de concurrencia utilizando WebSockets, lo cual permitió observar el comportamiento del sistema ante cargas simultáneas.

E. Optimización y propuestas de mejora

Con base en los resultados obtenidos, se formularon propuestas de mejora orientadas a incrementar la escalabilidad del sistema. Entre las tecnologías consideradas para futuras implementaciones se encuentran Firebase Cloud Messaging (FCM) y, eventualmente, sistemas de colas como RabbitMQ o Kafka, con el propósito de optimizar el procesamiento de grandes volúmenes de notificaciones.

Este enfoque metodológico permitió una evaluación ágil y sistemática de tecnologías para notificaciones en tiempo real, proporcionando una base sólida para futuras mejoras en términos de eficiencia, escalabilidad y robustez.

V. RESULTADOS

En este apartado se presentan los principales hallazgos derivados de la investigación sobre el diseño e implementación de una arquitectura para la gestión de notificaciones en tiempo real en la nube. El análisis se centró en variables críticas como la latencia, el costo por mensaje y el uso de recursos computacionales, comparando tecnologías existentes y poniendo estrategias para mejorar la eficiencia y escalabilidad del sistema como se muestra en la **Figura 2**.

Figura 2. Flujo de investigación de notificaciones en tiempo real en la nube.

Se identificaron oportunidades de mejora orientadas a la escalabilidad del sistema, tales como la implementación de técnicas de *batching* y la integración futura de servicios como *Firebase Cloud Messaging (FCM)*. Adicionalmente, se plantea la posibilidad de incorporar sistemas de colas de mensajes como *RabbitMQ o Kafka*, con el fin de mejorar la eficiencia en el procesamiento masivo de notificaciones.

La **Figura 3** ilustra el crecimiento lineal del costo total en función del número de notificaciones enviadas, con un costo fijo de 0.137 MXN por mensaje. Por ejemplo, enviar 100 notificaciones implica un costo de 13.65 MXN, cómo se resalta en la intersección de las líneas punteadas. Este análisis evidencia la relación proporcional entre el número de notificaciones y el costo total.

Figura 3. Costo total de notificaciones enviadas con Twilio.

La **Figura 4** muestra el aumento lineal de la latencia total con el número de notificaciones enviadas, con una latencia fija de 150 ms por mensaje. Por ejemplo, enviar 100 notificaciones genera una latencia acumulada de 15,000 ms, como se destaca en la intersección de las líneas punteadas. Este análisis permite visualizar cómo el incremento de notificaciones impacta el rendimiento del sistema.

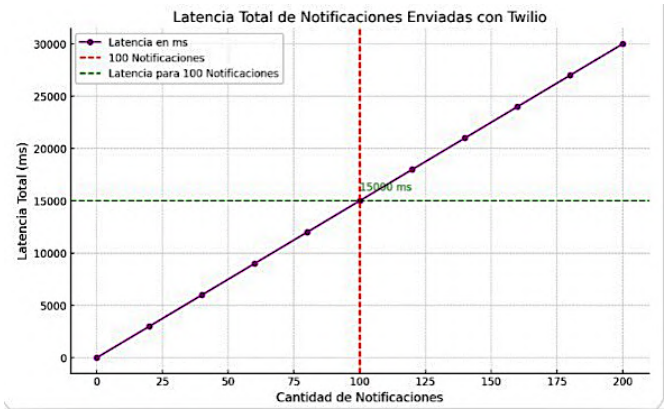

Figura 4. Latencia total de notificaciones enviadas con Twilio

Los resultados indican que el procesador Apple M1 es más eficiente que los Intel en el manejo de grandes volúmenes de datos, debido a la optimización de macOS y su arquitectura ARM. Sin embargo, futuras pruebas podrían demostrar un mejor desempeño en los Intel Core i9 o i7, optimizados para tareas de alta demanda computacional, como se observa en la **Figura 5**.

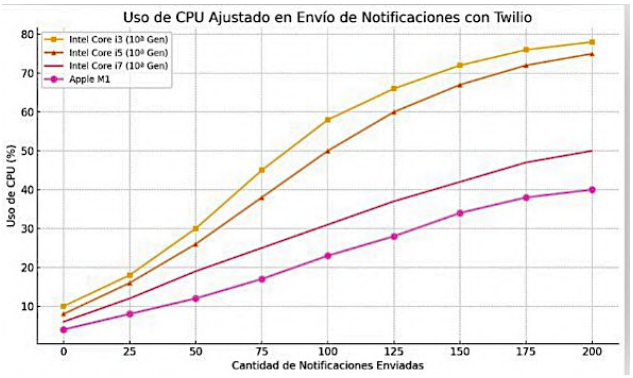

Figura 5. Uso de CPU en Twilio

La **Figura 6** muestra el aumento lineal de la latencia total en función de la cantidad de correos enviados, con una latencia fija de 50 ms por correo. Por ejemplo, enviar 100 correos genera una latencia acumulada de 5000 ms, como se destaca en la intersección de las líneas punteadas. Este análisis evidencia cómo el rendimiento del sistema se ve afectado por la acumulación de latencia a medida que crece el volumen de correos enviados.

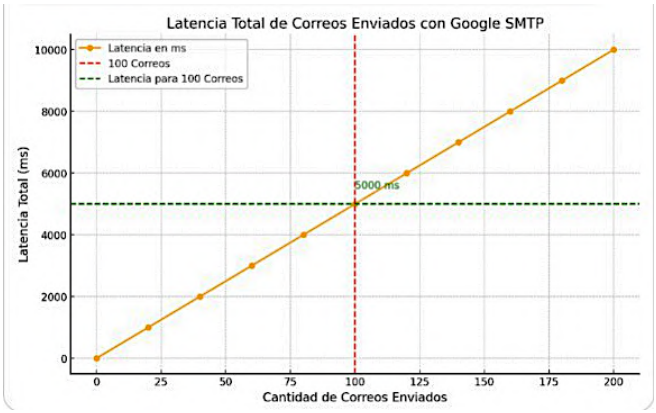

Figura 6. Latencia de correos con Google SMTP

La **Tabla 1** presenta un resumen de las características principales y los costos asociados a los planes de Gmail y Google Workspace. Se detalla el límite de correos electrónicos que se pueden enviar de manera gratuita por día, así como las opciones de suscripción disponibles para Google Workspace. Los costos se diferencian entre planes con compromiso anual y sin compromiso, permitiendo a los usuarios seleccionar la opción más adecuada según sus necesidades.

Tabla 1. Planes y límites de envío de correos en Gmail y Google Workspace.

Cuenta / Plan	Envios gratis por dia	Costo mensual (sin compromiso)
Gmail Gratuito	Hasta 100 correos	Gratis
Business Starter	Hasta 2,000 correos	Gratis
Business Standard	108 MXN por usuario	129,60 MXN por usuario
Business Plus	216 MXN por usuario	259,20 MXN por usuario
Enterprise	Contactar con ventas	Contactar con ventas

Fuente: googleworkspace2024

La **Figura 7** muestra la variación del uso de CPU en función de la cantidad de correos enviados mediante Google SMTP, comparando los procesadores Intel Core i3, i5, i7 y Apple M1. A medida que aumenta el número de correos, el consumo de CPU crece proporcionalmente, destacándose que los Intel Core i3 y i5 presentan un mayor consumo en comparación con el Apple M1, que resulta ser el más eficiente en la evaluación.

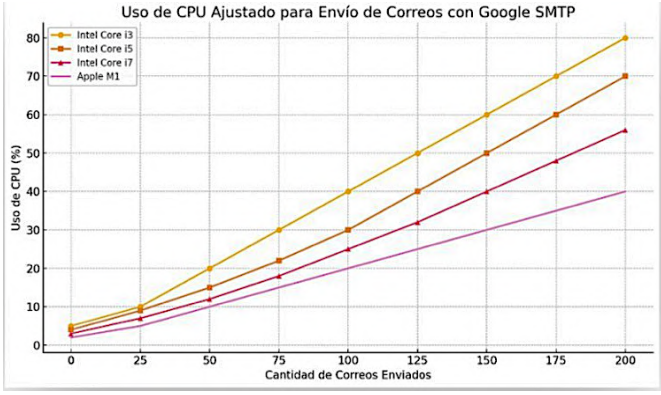

Figura 7. Uso de CPU Ajustado para Envío de Correos con Google SMTP

VI. CONCLUSIONES

Este trabajo evaluó de manera sistemática la latencia, el costo por mensaje y el consumo de CPU de dos servicios de notificación en la nube Twilio y Google SMTP) aplicando un proceso iterativo basado en Extreme Programming. Los experimentos muestran que Twilio mantiene una latencia media de 150 ms con un costo fijo de 0.137 MXN por SMS, mientras que Google SMTP reduce la latencia a 50 ms por correo, aunque está sujeto a cuotas de envío que limitan su escalabilidad inmediata.

En términos de hardware, el Apple M1 superó a los Intel Core i3 e i5 en eficiencia de CPU para ambas plataformas. Se sugiere ampliar las pruebas a los procesadores Intel Core i7 e i9 y a futuros Apple Silicon para verificar el comportamiento bajo cargas superiores a las analizadas.

Con estos hallazgos y recomendaciones, el estudio ofrece una guía práctica para seleccionar y dimensionar servicios de notificación en aplicaciones que demandan alto rendimiento, costos controlados y expansión futura.

REFERENCIAS

[1] D. F. Alfaro Quintero, D. M. Chaparro Macias, J. P. Lozano Novoa y J. D. Palma Roa, “Análisis de rendimiento de modelos gratuitos de Machine Learning (aprendizaje automático) utilizados en infraestructura de la nube en la predicción de la diabetes,” 2025.

[2] P. G. Uchuari Quiñónez, «Desarrollo de un sistema de gestión de citas y servicios para la peluquería “Shakinah”: frontend», Trabajo de Integración Curricular, Escuela Politécnica Nacional, Quito, Ecuador, 2024. [En línea]. Disponible en: http://bibdigital.epn.edu.ec/handle/15000/25582

[3] E. Trujillo Malaver, J. L. Daza Rojas y W. A. Morales Gallego, Propuesta de una mejora en el diseño de la arquitectura del sistema de gestión de la demanda WA Collaborative, Universidad Tecnológica, 2024. [En línea]. Disponible: http://hdl.handle.net/20.500.12622/6566

[4] P. Prabhakar, Y. Yuan, G. Yang, W. Sun, y A. Muralidharan, “Multi-objective optimization of notifications using offline reinforcement learning,” arXiv preprint arXiv:2207.03029, Jul. 2022. [En línea]. Disponible: https://arxiv.org/abs/2207.03029

[5] Y. Yuan, A. Muralidharan, P. Nandy, M. Cheng, y P. Prabhakar, “Offline reinforcement learning for mobile notifications,” arXiv preprint arXiv:2202.03867, Feb. 2022. [En línea]. Disponible: https://arxiv.org/abs/2202.03867

[6] N. Samarin, A. Sanchez, T. Chung, A. Bhavish Juleemun, C. Gilsenan, N. Merrill, J. Reardon, y S. Egelman, “The medium is the message: How secure messaging apps leak sensitive data to push notification services,” arXiv preprint arXiv:2407.10589v1, Jul. 2024. [En línea]. Disponible: https://arxiv.org/abs/2407.10589v1

BLANCA ESTELA ISLAS FLORES recibió su licenciatura en Ingeniería en Tecnologías de la Información y Comunicaciones del Tecnológico Nacional de México, campus Apizaco, en 2022. Posteriormente, ingresó a la Maestría en Sistemas Computacionales en el mismo plantel, con especialización en el área de desarrollo de software. A lo largo de su formación, Blanca ha destacado por su dedicación en la creación e implementación de soluciones tecnológicas innovadoras, participando activamente en diversos proyectos de desarrollo de software.
Entre sus actividades, ha participado en el diseño y desarrollo del Sistema de Gestión Integral de Servicios Dentales del Estado de Tlaxcala, su enfoque en la mejora continua y la innovación reflejan su compromiso con el avance de la tecnología en beneficio de la sociedad.

ARLEY IVÁN SOLIS ZACAPANTZI
Completó su Licenciatura en Ingeniería en Tecnologías de la Información y Comunicaciones en el Tecnológico Nacional de México, campus Apizaco, en 2022, y actualmente también cursa la Maestría en Sistemas Computacionales. A lo largo de su carrera, ha demostrado habilidades excepcionales en el desarrollo de software, participando en proyectos de gran relevancia junto con Blanca. Arley ha mostrado un fuerte compromiso en la investigación y aplicación de soluciones tecnológicas avanzadas.
Ambos autores trabajan de la mano en proyectos de investigación y desarrollo de software, con un enfoque en la innovación tecnológica para la optimización de sistemas en sectores clave, como los servicios dentales y la gestión de datos.

MARÍA JANAI SÁNCHEZ HERNÁNDEZ
Licenciada en Informática en el año 2001 y Maestra en Ciencias de la Computación en 2005, ambas carreras cursadas en el Instituto Tecnológico de Apizaco. Sus intereses académicos son la Ingeniería de Software, el uso de las metodologías ágiles, inteligencia artificial aplicada a la ingeniería de software, la deuda técnica y el desarrollo de software. Desde 2004 es docente del área de Sistemas y Computación del TecNM Apizaco impartiendo diversas asignaturas, y desde 2015 es colaboradora de la Maestría en Sistemas Computacionales del Instituto Tecnológico de Apizaco.

JOSÉ JUAN HERNÁNDEZ MORA
Es Ingeniero en Computación por la Universidad Autónoma de Tlaxcala. Tiene el grado de Maestro en Ciencias en Computacional es por el Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET), de Cuernavaca, Morelos y Doctor en Excelencia Docente por la Universidad de los Ángeles. Es Profesor con Perfil Deseable por parte del PRODEP, es líder del cuerpo académico "Sistemas de Información" y nivel de candidato del SNII del CONAHCYT. Sus líneas de investigación incluyen: Ingeniería de Software, Desarrollo de Aplicaciones de Tecnologías de la Información, Procesamiento Digital de Imágenes (PDI), Redes Neuronales Artificiales (RNA).

JUAN RAMOS RAMOS
Licenciado en Informática por el Instituto Tecnológico de Apizaco, con estudios de Maestría en Ciencias Computacionales y Telecomunicaciones por el Instituto de Estudios Universitarios, A.C. y Doctorado en Sistemas Computacionales por la Universidad del Sur. PTC en el Tecnológico Nacional de México (TecNM) Instituto Tecnológico de Apizaco. Catedra a nivel licenciatura en la carrera de Ingeniería en Tecnologías de la Información y Comunicaciones, y a nivel posgrado, en la Maestría en Sistemas Computacionales.
Miembro del Cuerpo Académico "Sistemas de Información" con clave ITAPI-CA-6. Reconocimiento a Perfil Deseable otorgado por el PRODEP, a partir del año 2019.
Principales áreas de interés: Bases de datos, Ingeniería de Software, Sistemas distribuidos y Cómputo en la nube.
Desarrollo de proyectos de investigación y desarrollo tecnológico que solucionen necesidades en las organizaciones, liderando proyectos con el Tribunal Superior de Justicia del Estado de Tlaxcala, con el Municipio de Tzompantepec, con el Instituto Nacional de Migración y con la Asociación Dental del Estado de Tlaxcala, entre otras organizaciones del sector público.

ELIZABETH CUATECONTZI CUAHUTLE
Es Licenciada en Informática por el Instituto Tecnológico de Apizaco. Maestra en Dirección de Ingeniería de Software por el Instituto de Estudios Universitarios. Docente de tiempo completo en el Depto. de Sistemas y Computación del Tecnológico Nacional de México Campus Apizaco. Miembro del consejo de la Maestría en Sistemas Computacionales del TecNM Campus Apizaco. Área de interés: ingeniería de software y sistemas distribuidos. Actualmente realiza los estudios de doctorado en el programa de doctorado en ciencias de la ingeniería.