UNIVERSIDAD NACIONAL DE SAN ANTONIO ABAD DEL CUSCO

APLICACIÓN DE LA MINERÍA DE PROCESOS PARA MEJORAR LA MIGRACIÓN DE ARQUITECTURAS MONOLÍTICAS A MICROSERVICIOS

Trabajo de investigación presentado por:
Callapiña Castilla Ciro Gabriel
134403
Zegarra Rojas Jorge Enrique
161534

Bajo la asesoría de:
Yeshica Isela Ormeño Ayala

Perú, Abril de 2025

Escuela Profesional de Ingeniería Informática y de Sistemas
Facultad de Ingeniería Eléctrica, Electrónica, Informática y Mecánica
Universidad Nacional de San Antonio Abad del Cusco

Índice general

1. INTRODUCCIÓN

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

2. JUSTIFICACIÓN DE LA INVESTIGACIÓN

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

3. PLANTEAMIENTO Y FORMULACIÓN DEL PROBLEMA DE INVESTIGACIÓN

3.1 PLANTEAMIENTO DEL PROBLEMA

3.1.1 Contexto: El Dilema de las Arquitecturas Monolíticas y el Auge de los Microservicios

Las arquitecturas de software monolíticas, caracterizadas por compilar una aplicación como una única unidad autocontenida (Eski y Buzluca, 2018), han sido durante mucho tiempo un estándar en el desarrollo de software. Su simplicidad inicial facilita el desarrollo, las pruebas integrales y el despliegue en las primeras etapas de un proyecto. Sin embargo, a medida que estas aplicaciones crecen en tamaño y complejidad funcional, sus inherentes limitaciones se vuelven cada vez más evidentes y problemáticas (Eski y Buzluca, 2018).

El mantenimiento de grandes bases de código monolíticas se torna progresivamente más desafiante, afectando negativamente la productividad de los equipos de desarrollo y la calidad del software entregado. Además, la escalabilidad de las aplicaciones monolíticas es limitada, ya que requiere escalar toda la aplicación en conjunto, independientemente de los componentes específicos que experimenten mayor demanda. Esta situación genera ineficiencias significativas en el uso de recursos computacionales.

Frente a estos desafíos, la arquitectura de microservicios ha surgido como un enfoque alternativo prometedor. Al dividir una aplicación en un conjunto de servicios independientes que se comunican entre sí, los microservicios ofrecen numerosos beneficios: escalabilidad granular, despliegues más ágiles y frecuentes, mayor resiliencia ante fallos, y flexibilidad tecnológica al permitir la adopción de diferentes lenguajes y herramientas para distintos servicios (Abgaz y cols., 2023). Estos atributos han posicionado a los microservicios como una elección atractiva para aplicaciones que requieren alta disponibilidad, escalabilidad y capacidad de evolución rápida.

3.1.2 El Desafío Crítico: La Migración y Descomposición del Monolito

A pesar de los atractivos beneficios, la transición de una arquitectura monolítica a una de microservicios representa un desafío considerable para las organizaciones. Esta migración implica la descomposición cuidadosa de un sistema altamente acoplado en servicios independientes, respetando las dependencias funcionales y de datos existentes (Li, Ma, y Lu, 2020).

Una descomposición inapropiada puede resultar en arquitecturas frágiles, con dependencias ocultas entre microservicios, duplicación innecesaria de lógica de negocio, e inconsistencias en la gestión de datos. Estos problemas pueden derivar en una arquitectura de microservicios subóptima, difícil de mantener y escalar, socavando los beneficios esperados de la transición (Taibi y Systä, 2019).

La complejidad de la migración se ve exacerbada en aplicaciones legadas de gran tamaño, donde el conocimiento del sistema puede estar incompleto o desactualizado, y donde la documentación puede ser insuficiente. En estos casos, comprender la estructura funcional y las interacciones internas del monolito es un requisito previo esencial para una descomposición exitosa.

3.1.3 Limitaciones de los Enfoques Actuales y la Oportunidad de la Minería de Procesos

Si bien los enfoques existentes para la descomposición de monolitos aportan valor, también presentan limitaciones inherentes que pueden comprometer la calidad de la arquitectura de microservicios resultante.

Las estrategias manuales dependen en gran medida del conocimiento tácito de los expertos en el sistema, lo que introduce riesgos de subjetividad, sesgos y pérdida de información crítica (Hasan, Osman, Admodisastro, y Muhammad, 2023). Por otro lado, los enfoques automatizados basados en análisis estático del código, aunque más objetivos, tienden a centrarse únicamente en las dependencias estructurales, sin capturar adecuadamente los flujos de trabajo dinámicos y las interacciones reales en tiempo de ejecución (Matias y cols., 2020; Levezinho, Kapferer, Zimmermann, y Silva, 2024).

La Minería de Procesos (Process Mining) surge como una oportunidad prometedora para superar estas limitaciones. Esta disciplina permite analizar de manera sistemática los registros de eventos generados por los sistemas de software para descubrir, monitorear y mejorar los procesos reales de negocio y operativos (Aguirre Mayorga y Rincón García, 2015).

Al aplicar técnicas de minería de procesos a los monolitos, es posible inferir modelos de comportamiento que reflejan cómo se utilizan realmente las funcionalidades,

cómo fluyen los datos y cómo interactúan los distintos módulos en escenarios reales de ejecución. Esta perspectiva basada en la evidencia complementa y enriquece los enfoques tradicionales de descomposición.

3.2 FORMULACIÓN DEL PROBLEMA

Ante las limitaciones de los enfoques actuales y el potencial demostrado por la Minería de Procesos para analizar el comportamiento real de los sistemas a través de sus logs de eventos, surge la siguiente cuestión central de investigación:

> *¿Cómo puede aprovecharse la Minería de Procesos para guiar de manera más efectiva la descomposición de arquitecturas monolíticas en arquitecturas de microservicios, mejorando así la calidad de la migración y maximizando los beneficios esperados de escalabilidad, mantenibilidad y resiliencia?*

Esta pregunta orienta la presente investigación hacia el diseño, implementación y validación de un enfoque basado en minería de procesos que asista en la descomposición de sistemas monolíticos, buscando aportar una solución más fundamentada, objetiva y efectiva a este desafío crítico en la evolución de sistemas de software.

4. OBJETIVOS

4.1 OBJETIVO GENERAL

Aplicar la minería de procesos para mejorar la migración de arquitecturas monolíticas a microservicios.

4.2 OBJETIVOS ESPECÍFICOS

- Analizar los desafíos y enfoques existentes en la descomposición de aplicaciones monolíticas para su migración hacia arquitecturas de microservicios.
- Identificar y adaptar técnicas de Minería de Procesos que permitan analizar el comportamiento operativo de sistemas monolíticos con el fin de descubrir candidatos a microservicios.
- Diseñar una estrategia metodológica sistemática que integre la Minería de Procesos para la identificación de microservicios y la definición precisa de sus límites.
- Establecer criterios y métricas de evaluación que permitan medir la efectividad, calidad y robustez de las descomposiciones obtenidas mediante la estrategia propuesta

5. MARCO TEÓRICO

5.1 ANTECEDENTES

A continuación, se presentan estudios destacados que proponen metodologías y herramientas concretas para facilitar este proceso, abordando desde el enfoque en minería de proceso con trabajos tempranos hasta análisis automatizado de código y trazas de ejecución.

En un estudio de Taibi y Systä en 2019, titulado *From Monolithic Systems to Microservices: A Decomposition Framework based on Process Mining*, se propone un framework de 6 pasos que usa minería de procesos para ayudar a dividir un sistema monolítico en microservicios (Taibi y Systä, 2019). Esto lo hicieron en Finlandia, en el grupo de investigación Tampere Software Engineering. Básicamente, agarran los logs de ejecución del sistema, los analizan con una herramienta llamada DISCO, y de ahí sacan rutas de ejecución que se repiten mucho. Lo probaron con un caso real de una empresa: el sistema daba opciones de cómo partir el monolito, y esas opciones se compararon con lo que el arquitecto de software había hecho a mano. Usaron trazas de logs, la herramienta DISCO, y unas métricas para evaluar todo. Lo interesante fue que el framework encontró problemas en la arquitectura que el análisis manual no había visto, y además propuso más formas de dividir el sistema que las que el arquitecto tenía pensadas. Como conclusión, todo esto mejora mucho el proceso porque da más opciones y hace que las decisiones no dependan tanto de la intuición del arquitecto.

Un año después, Taibi y Systä también presentaron otro trabajo en la conferencia CLOSER 2020, titulado *A Decomposition and Metric-Based Evaluation Framework for Microservices*, que básicamente complementa lo anterior (Taibi y Systä, 2020). En este estudio propusieron dos cosas: un sistema de métricas para comparar distintas arquitecturas resultantes, y otro proceso de descomposición que también se basa en minería de procesos. Aunque no trabajaron con una población específica, sí mostraron cómo sus herramientas sirven para extraer candidatos a microservicios desde un monolito. Otra vez usaron logs de ejecución y herramientas que sacan métricas arquitectónicas de forma automática. Este método ayudó a encontrar distintas formas de dividir el sistema y a reducir lo subjetivo que puede ser elegir cómo hacerlo. También permite ver si con el tiempo la arquitectura se empieza a deteriorar. En conclusión, si combinás minería de

procesos con métricas, se puede tomar decisiones más objetivas a la hora de migrar de un monolito a microservicios.

En una investigación más reciente de Zougari y otros autores en 2024, llamada *Automating the Recognition of Microservices from Business Process Analysis*, se propone una forma de extraer microservicios automáticamente a partir de modelos de procesos de negocio, usando minería de procesos para analizar flujos de trabajo (por ejemplo, diagramas BPMN) y agrupar actividades que tienen sentido juntas (Zougari, Daoud, Abdelouahed, y Zougari, 2024). Aunque no trabajaron directamente con sistemas legados, probaron el enfoque con casos reales y mostraron que se puede derivar un conjunto de microservicios que se ajusta a lo que el negocio necesita. Todo esto acelera bastante la migración del monolito porque se basa en cómo fluye realmente el trabajo, no en cómo está hecho el código por dentro.

Otro trabajo interesante es el de Klewerton y su equipo en 2021, titulado *Modernizing Legacy Systems with Microservices: A Roadmap*, donde hacen una revisión sistemática sobre migración a microservicios (Wolfart y cols., 2021). Aunque no se enfocan únicamente en minería de procesos, mencionan que hay muy pocos estudios con herramientas prácticas, y destacan explícitamente el framework de Taibi y Systä de 2019 como una de las pocas propuestas útiles. Lo que encontraron es que la mayoría de migraciones todavía se hacen a mano y falta automatización. En ese contexto, remarcan que la minería de procesos puede sacar procesos de negocio directamente de los logs y dar sugerencias sobre cómo dividir el sistema. Este trabajo no aporta resultados nuevos, pero confirma que los enfoques que usan minería de procesos están bien fundamentados y validados en otros estudios.

En el trabajo de (Ren y cols., 2018) llamado *Migrating Web Applications from Monolithic Structure to Microservices Architecture*, unos investigadores de China crearon un enfoque interesante que mezcla análisis estático y dinámico para entender bien una aplicación monolítica. Básicamente lo que hicieron fue extraer el grafo de llamadas a funciones mientras la app estaba corriendo y medir qué tan acopladas estaban las funciones entre sí. Después aplicaron clustering sobre las funciones (basándose en cómo dependían unas de otras) para sugerir qué partes deberían convertirse en microservicios. Para probar que su método funcionaba, migraron una aplicación web legada "típica" a microservicios, y demostraron que su técnica identificaba los límites de cada servicio con bastante exactitud y sin consumir demasiados recursos. Sus resultados fueron bastante buenos: lograron crear servicios cohesivos y con poco acoplamiento gracias al clustering de funciones. Al final concluyeron que combinar análisis estático, o sea, ver cómo está estructurado el código sin ejecutarlo y dinámico que es observar trazas de ejecución real permite generar automáticamente sugerencias de particiones eficientes para microservicios.

(Ding, Peng, Guo, Zhang, y Wu, 2020), titulado *Scenario-driven and Bottom-up*

Microservice Decomposition Method for Monolithic Systems, un grupo de la Universidad Fudan en Shanghai, desarrollaron un método para automatizar la descomposición de un monolito en microservicios, pero enfocándose especialmente en cómo partir la base de datos y el código. Para lograrlo, usaron análisis dinámico con una herramienta llamada Kieker que recoge trazas de llamadas a métodos y operaciones de base de datos mientras el sistema está funcionando. Con esa info generaron un "grafo de trayectoria de acceso a datos", suena complicado pero es básicamente ver cómo se accede a los datos, analizaron cómo se relacionan las tablas entre sí. A partir de ahí, calcularon dinámicamente diferentes formas de partir los datos y, desde abajo hacia arriba, propusieron cómo agrupar los módulos de código asociados a cada partición de datos. Hasta desarrollaron una herramienta prototipo super útil llamada MSDecomposer que muestra visualmente todo el proceso y te deja ajustar manualmente la solución si no te convence. Cuando probaron su método en varios sistemas open-source, vieron que aceleraba muchísimo la toma de decisiones de partición, o sea, reducía el trabajo manual y producía particiones razonables tanto de la base de datos como del código. El resultado final era una descomposición con buena cohesión funcional y bajo acoplamiento. Su conclusión fue que este enfoque guiado por escenarios de uso reales, con el apoyo del análisis dinámico, hace más confiable y rápido el diseño de microservicios a partir de un monolito, y le quita un montón de presión a los desarrolladores que antes tenían que tomar decisiones más subjetivas.

En otro estudio, (Haferkorn, Kerth, Rodenbeck, y Zaschke, 2020) titulado *Migration from Monolith to Microservices with Legacy Compatibility*, hicieron un estudio de caso interesante sobre un sistema monolítico militar que usaba middleware viejo y obsoleto (CORBA), y lo adaptaron usando Domain-Driven Design y estrategias ágiles. Su objetivo era reconstruir el sistema en componentes más pequeños pero manteniendo compatibilidad con las interfaces antiguas, para que todo siguiera funcionando mientras se hacía la migración. Para conseguirlo, redefinieron el dominio en "contextos acotados" para reorganizar mejor el código, adoptaron prácticas ágiles (usaron Scrum pero adaptado a sus necesidades) y sacaron algunas funcionalidades como la autenticación para convertirlas en servicios independientes. Como resultado, lograron diseñar y probar exitosamente un microservicio para la lógica de autenticación que podía reutilizarse, y lo pusieron en producción. Notaron que incluso esta migración parcial ya traía mejoras en la mantenibilidad y escalabilidad del sistema. Su conclusión fue que, aunque hicieron el proceso de forma incremental, el uso de Domain-Driven Design les ayudó montón a dividir el monolito según la lógica de negocio, y su pequeño equipo pudo aplicar nuevas prácticas para apoyar la migración. Una lección importante que aprendieron es que para hacer una reingeniería efectiva se necesitan cambios en la cultura de trabajo del equipo, no solo adoptar patrones técnicos de microservicios.

En el trabajo de (Kalia y cols., 2021) titulado *Mono2Micro: a practical and*

effective tool for decomposing monolithic Java applications to microservices, un equipo de IBM Research en Estados Unidos, desarrollaron Mono2Micro, una herramienta basada en IA para ayudar a los ingenieros a refactorizar aplicaciones monolíticas. Esta herramienta recolecta información estática pero sobre todo se enfoca en trazas de ejecución de casos de uso empresariales bien definidos. Lo que hace es aplicar un enfoque de descomposición "espacio-temporal": el espacio lo forman los casos de uso (para asegurar cohesión funcional), y el tiempo son las relaciones de llamadas directas e indirectas que se extraen de las trazas. Después aplica clustering jerárquico de clases Java que aparecen en las trazas para proponer particiones que podrían convertirse en microservicios. Probaron la herramienta con varias aplicaciones Java, tanto open source como propietarias, e incluso hicieron comparativas con lo que se considera la partición "correcta" en ejemplos conocidos como Spring PetClinic. Al comparar con otras cuatro técnicas existentes, Mono2Micro obtuvo mejores resultados en las métricas de calidad de descomposición y eficiencia de la herramienta. Además, hicieron una encuesta a 21 profesionales que mostró que los desarrolladores ven beneficios claros cuando usan la herramienta, especialmente porque genera servicios cohesionados y fáciles de entender. Concluyeron que Mono2Micro facilita muchísimo la migración gracias a sus recomendaciones automáticas: segmenta el dominio en servicios cohesivos y explica los límites de negocio de cada servicio, lo que la convierte en una ayuda súper práctica para diseñar arquitecturas de microservicios en entornos industriales reales.

En el trabajo de (Würz, Krämer, Kaster, y Kuijper, 2023), aunque está más enfocado en FaaS (Function as a Service, o serverless), proponen un método técnico que también sirve para microservicios. Su objetivo era guiar cómo partir un monolito en funciones pequeñas para entornos serverless. Para lograrlo, plantean un proceso de tres pasos: primero, identificar las tareas principales de la aplicación y sus subtareas; segundo, definir el flujo del programa para detectar qué tareas podrían ser funciones autónomas y cómo interactuarían entre ellas; y tercero, especificar funciones concretas y, si es necesario, combinar aquellas que sean demasiado pequeñas (para evitar tener demasiada sobrecarga de comunicación). Evaluaron su método con un caso real, una aplicación de almacenamiento de datos geoespaciales. Los resultados mostraron que su metodología produce funciones independientes del tamaño adecuado para una ejecución eficiente en entornos FaaS. Su conclusión fue que este enfoque estructurado ayuda a diseñar particiones lógicas, ya sean funciones o microservicios bien balanceadas en términos de complejidad y comunicación, lo que facilita modernizar incrementalmente un monolito hacia una arquitectura de servicios más manejables.

En el trabajo de Trabelsi et al. (Trabelsi y cols., 2022) con el título *From legacy to microservices: A type-based approach for microservices identification using machine learning and semantic analysis*, se propuso un método llamado MicroMiner, cuyo objetivo principal fue identificar microservicios en sistemas legados utilizando una combi-

nación de análisis estático de código, análisis semántico y aprendizaje automático. Los autores trabajaron con cuatro sistemas monolíticos reales como muestra y desarrollaron un enfoque que incluía relaciones entre elementos del código fuente, embeddings para análisis semántico, y una clasificación basada en tipos de servicio. Usaron tanto evaluaciones cualitativas como cuantitativas para medir los resultados. El método logró una precisión del 68.15 % y un *recall* del 77 %, lo cual demuestra buenos resultados para este tipo de tareas complejas. La conclusión del estudio fue que MicroMiner puede automatizar con éxito la etapa de identificación de servicios, generando microservicios significativos desde el punto de vista arquitectónico, y que es una alternativa válida para apoyar la migración.

En el trabajo de Filippone et al. (Filippone, Mehmood, Autili, Rossi, y Tivoli, 2023) titulado *From monolithic to microservice architecture: An automated approach based on graph clustering and combinatorial optimization*, el objetivo fue automatizar la identificación de microservicios usando técnicas de agrupación de grafos y optimización combinatoria. Para eso, analizaron cuatro sistemas monolíticos y aplicaron análisis estático del código para construir un grafo de dependencias. Luego, usaron el algoritmo Louvain para detectar comunidades con alta cohesión y aplicaron algoritmos de optimización para reducir el acoplamiento entre servicios. La evaluación usó métricas específicas de cohesión y acoplamiento, y los resultados mostraron que su método generó arquitecturas bastante cohesionadas, superando enfoques previos. En su conclusión, los autores destacan que este enfoque ofrece soporte automatizado y efectivo para la migración arquitectónica, con resultados prometedores que podrían servir en aplicaciones reales.

En el trabajo de Rochimah y Nuralamsyah (Rochimah y Nuralamsyah, 2023) titulado *Decomposing Monolithic to Microservices: Keyword Extraction and BFS Combination Method*, se propuso un enfoque algo más simple pero interesante, que extrae palabras clave del código fuente y usa un recorrido en anchura (BFS) para agrupar clases o componentes del monolito en posibles microservicios. El estudio fue aplicado sobre una aplicación monolítica concreta y se comparó su método contra otro método alternativo. El resultado fue que su enfoque alcanzó una precisión promedio de 0.81, lo cual es bastante alto en este contexto. En la conclusión, los autores afirman que su método mejora la exactitud en la identificación de microservicios, especialmente cuando se basa en el análisis del código fuente, aunque también reconocen que se necesita más investigación para generalizarlo.

En el trabajo de Martínez Saucedo y Rodríguez (Martínez Saucedo y Rodríguez, 2024) titulado *Migration of Monolithic Systems to Microservices using AI: A Systematic Mapping Study*, se presentó otro estudio de mapeo sistemático, pero enfocado específicamente en el uso de técnicas de Inteligencia Artificial para la migración. El objetivo fue caracterizar el estado actual de la investigación que aplica IA en este campo. Se

seleccionaron 22 estudios relevantes y se observó que el 63 % utiliza *clustering* como técnica principal, mientras que el 36.4 % emplea el código fuente como entrada principal. En su conclusión, se afirma que el *clustering* basado en código es la tendencia dominante, lo cual sugiere un movimiento hacia métodos automáticos de descomposición del monolito con IA.

Por último, en el trabajo de Lecrivain et al. (Lecrivain, Barry, Tamzalit, y Sahraoui, 2025) titulado *MONO2REST: Identifying and Exposing Microservices: a Reusable RESTification Approach*, se propuso MONO2REST, un método que no busca migrar todo el sistema, sino exponer microservicios directamente desde un monolito usando algoritmos genéticos y clasificadores para generar *endpoints* REST. El estudio se basó en la conocida aplicación Spring PetClinic, comparando la versión monolítica con su versión en microservicios. El enfoque logró identificar microservicios alineados con la implementación de referencia, lo que valida su utilidad. En su conclusión, se indica que MONO2REST permite migraciones parciales sin reescribir todo el sistema, lo cual puede ser muy útil en contextos donde no es factible una transformación completa.

Los trabajos revisados muestran que la minería de procesos tiene un gran potencial en la migración de sistemas monolíticos a microservicios, ya que permite descubrir cómo se comporta el sistema en la práctica mediante el análisis de logs y trazas de ejecución. Esto ayuda a identificar componentes conectados, rutas frecuentes y posibles divisiones que no siempre se detectan solo leyendo el código. Estudios como los de Taibi y Systä destacan que combinar estos análisis con métricas permite tomar decisiones más objetivas al diseñar microservicios.

Aunque aún hay poca automatización y se necesita más validación en escenarios reales, los resultados son prometedores. Además, muchos enfoques combinan la minería de procesos con técnicas como clustering, algoritmos genéticos o análisis estático, mostrando que no existe una única solución válida para todos los casos. La elección depende del contexto y los objetivos del proyecto. Sin embargo, todos coinciden en que, bien aplicada, esta migración mejora la modularidad, escalabilidad y orden de los sistemas legados.

5.2 BASES TEÓRICAS

Etiam ac leo a risus tristique nonummy. Donec dignissim tincidunt nulla. Vestibulum rhoncus molestie odio. Sed lobortis, justo et pretium lobortis, mauris turpis condimentum augue, nec ultricies nibh arcu pretium enim. Nunc purus neque, placerat id, imperdiet sed, pellentesque nec, nisl. Vestibulum imperdiet neque non sem accumsan laoreet. In hac habitasse platea dictumst. Etiam condimentum facilisis libero. Suspendisse in elit quis nisl aliquam dapibus. Pellentesque auctor sapien. Sed egestas sapien nec lectus. Pellentesque vel dui vel neque bibendum viverra. Aliquam porttitor nisl nec

pede. Proin mattis libero vel turpis. Donec rutrum mauris et libero. Proin euismod porta felis. Nam lobortis, metus quis elementum commodo, nunc lectus elementum mauris, eget vulputate ligula tellus eu neque. Vivamus eu dolor.

Nulla in ipsum. Praesent eros nulla, congue vitae, euismod ut, commodo a, wisi. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Aenean nonummy magna non leo. Sed felis erat, ullamcorper in, dictum non, ultricies ut, lectus. Proin vel arcu a odio lobortis euismod. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Proin ut est. Aliquam odio. Pellentesque massa turpis, cursus eu, euismod nec, tempor congue, nulla. Duis viverra gravida mauris. Cras tincidunt. Curabitur eros ligula, varius ut, pulvinar in, cursus faucibus, augue.

Nulla mattis luctus nulla. Duis commodo velit at leo. Aliquam vulputate magna et leo. Nam vestibulum ullamcorper leo. Vestibulum condimentum rutrum mauris. Donec id mauris. Morbi molestie justo et pede. Vivamus eget turpis sed nisl cursus tempor. Curabitur mollis sapien condimentum nunc. In wisi nisl, malesuada at, dignissim sit amet, lobortis in, odio. Aenean consequat arcu a ante. Pellentesque porta elit sit amet orci. Etiam at turpis nec elit ultricies imperdiet. Nulla facilisi. In hac habitasse platea dictumst. Suspendisse viverra aliquam risus. Nullam pede justo, molestie nonummy, scelerisque eu, facilisis vel, arcu.

5.3 DEFINICIÓN DE TÉRMINOS BÁSICOS

Curabitur tellus magna, porttitor a, commodo a, commodo in, tortor. Donec interdum. Praesent scelerisque. Maecenas posuere sodales odio. Vivamus metus lacus, varius quis, imperdiet quis, rhoncus a, turpis. Etiam ligula arcu, elementum a, venenatis quis, sollicitudin sed, metus. Donec nunc pede, tincidunt in, venenatis vitae, faucibus vel, nibh. Pellentesque wisi. Nullam malesuada. Morbi ut tellus ut pede tincidunt porta. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam congue neque id dolor.

6. FORMULACIÓN DE HIPÓTESIS

6.1 HIPÓTESIS GENERAL

Donec et nisl at wisi luctus bibendum. Nam interdum tellus ac libero. Sed sem justo, laoreet vitae, fringilla at, adipiscing ut, nibh. Maecenas non sem quis tortor eleifend fermentum. Etiam id tortor ac mauris porta vulputate. Integer porta neque vitae massa. Maecenas tempus libero a libero posuere dictum. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aenean quis mauris sed elit commodo placerat. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Vivamus rhoncus tincidunt libero. Etiam elementum pretium justo. Vivamus est. Morbi a tellus eget pede tristique commodo. Nulla nisl. Vestibulum sed nisl eu sapien cursus rutrum.

6.2 HIPÓTESIS ESPECÍFICAS

- Nulla non mauris vitae wisi posuere convallis. Sed eu nulla nec eros scelerisque pharetra. Nullam varius.
- Etiam dignissim elementum metus. Vestibulum faucibus, metus sit amet mattis rhoncus, sapien dui laoreet odio, nec ultricies nibh augue a enim. Fusce in ligula.
- Quisque at magna et nulla commodo consequat. Proin accumsan imperdiet sem. Nunc porta.

7. DISEÑO METODOLÓGICO

7.1 TIPO Y NIVEL DE INVESTIGACIÓN

Nulla ac nisl. Nullam urna nulla, ullamcorper in, interdum sit amet, gravida ut, risus. Aenean ac enim. In luctus. Phasellus eu quam vitae turpis viverra pellentesque. Duis feugiat felis ut enim. Phasellus pharetra, sem id porttitor sodales, magna nunc aliquet nibh, nec blandit nisl mauris at pede. Suspendisse risus risus, lobortis eget, semper at, imperdiet sit amet, quam. Quisque scelerisque dapibus nibh. Nam enim. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nunc ut metus. Ut metus justo, auctor at, ultrices eu, sagittis ut, purus. Aliquam aliquam.

7.2 POBLACIÓN Y MUESTRA

Etiam pede massa, dapibus vitae, rhoncus in, placerat posuere, odio. Vestibulum luctus commodo lacus. Morbi lacus dui, tempor sed, euismod eget, condimentum at, tortor. Phasellus aliquet odio ac lacus tempor faucibus. Praesent sed sem. Praesent iaculis. Cras rhoncus tellus sed justo ullamcorper sagittis. Donec quis orci. Sed ut tortor quis tellus euismod tincidunt. Suspendisse congue nisl eu elit. Aliquam tortor diam, tempus id, tristique eget, sodales vel, nulla. Praesent tellus mi, condimentum sed, viverra at, consectetuer quis, lectus. In auctor vehicula orci. Sed pede sapien, euismod in, suscipit in, pharetra placerat, metus. Vivamus commodo dui non odio. Donec et felis.

7.3 TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE DATOS

Etiam suscipit aliquam arcu. Aliquam sit amet est ac purus bibendum congue. Sed in eros. Morbi non orci. Pellentesque mattis lacinia elit. Fusce molestie velit in ligula. Nullam et orci vitae nibh vulputate auctor. Aliquam eget purus. Nulla auctor wisi sed ipsum. Morbi porttitor tellus ac enim. Fusce ornare. Proin ipsum enim, tincidunt in, ornare venenatis, molestie a, augue. Donec vel pede in lacus sagittis porta. Sed hendrerit ipsum quis nisl. Suspendisse quis massa ac nibh pretium cursus. Sed sodales. Nam eu

neque quis pede dignissim ornare. Maecenas eu purus ac urna tincidunt congue.

7.4 TÉCNICAS DE PROCESAMIENTO Y ANÁLISIS DE DATOS

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

8. CRONOGRAMA DE ACTIVIDADES

Actividad	**Mes 1**	**Mes 2**	**Mes 3**	**Mes 4**	**Mes 5**	**Mes 6**
Revisión bibliográfica	X	X				
Elaboración del marco teórico		X	X			
Diseño de instrumentos			X	X		
Recolección de datos				X	X	
Análisis de resultados					X	X
Elaboración del informe final						X

9. PRESUPUESTO

Rubro	**Cantidad**	**Costo (S/.)**
Material de escritorio	-	500.00
Equipo informático	-	3000.00
Servicios de impresión	-	300.00
Movilidad	-	500.00
Otros gastos	-	700.00
Total		**5000.00**

REFERENCIAS

Abgaz, Y., McCarren, A., Elger, P., Solan, D., Lapuz, N., Bivol, M., y Clarke, P. (2023). Decomposition of monolith applications into microservices architectures: A systematic review. *IEEE Transactions on Software Engineering*, *49*(8), 4213–4242.

Aguirre Mayorga, H. S., y Rincón García, N. (2015). Minería de procesos: desarrollo, aplicaciones y factores críticos. *Cuadernos de Administración*, *28*(50), 137–157.

Ding, D., Peng, X., Guo, X., Zhang, J., y Wu, Y. (2020). Scenario-driven and bottom-up microservice decomposition method for monolithic systems. *Journal of Software*, *31*(11), 3461–3480. ([in Chinese])

Eski, S., y Buzluca, F. (2018). An automatic extraction approach: Transition to microservices architecture from monolithic application. En *Proceedings of the 19th international conference on agile software development: Companion* (pp. 1–6).

Filippone, G., Mehmood, N. Q., Autili, M., Rossi, F., y Tivoli, M. (2023). From monolithic to microservice architecture: An automated approach based on graph clustering and combinatorial optimization. En *Proceedings of the 2023 ieee international conference on software architecture (icsa 2023).*

Haferkorn, D., Kerth, C., Rodenbeck, R., y Zaschke, C. (2020). Migration from monolith to microservices with legacy compatibility. En *Publicado en actas de congreso.* Fraunhofer IOSB, Karlsruhe, Alemania.

Hasan, M. H., Osman, M. H., Admodisastro, N. I., y Muhammad, M. S. (2023). A quality driven framework for decomposing legacy monolith applications to microservice architecture.
(Preprint)

Kalia, A. K., Xiao, J., Krishna, R., Sinha, S., Vukovic, M., y Banerjee, D. (2021). Mono2micro: a practical and effective tool for decomposing monolithic java applications to microservices. En *Proc. of esec/fse 2021.* Athens, Grecia.

Lecrivain, M., Barry, H., Tamzalit, D., y Sahraoui, H. (2025). Mono2rest: Identifying and exposing microservices: a reusable restification approach. *arXiv preprint arXiv:2503.21522*.

Levezinho, M., Kapferer, S., Zimmermann, O., y Silva, A. R. (2024). Domain-driven design representation of monolith candidate decompositions based on entity accesses. *arXiv preprint arXiv:2407.02512*.

Li, C. Y., Ma, S. P., y Lu, T. W. (2020). Microservice migration using strangler fig pattern: A case study on the green button system. En *2020 international computer symposium (ics)* (pp. 519–524).

Martínez Saucedo, A. C., y Rodríguez, G. H. (2024). Migration of monolithic systems to microservices using ai: A systematic mapping study. En *Anais do xxvii congresso ibero-americano em engenharia de software (cibse 2024).* (Art. 28435)

Matias, T., Correia, F. F., Fritzsch, J., Bogner, J., Ferreira, H. S., y Restivo, A. (2020). Determining microservice boundaries: A case study using static and dynamic software analysis. En *Software architecture: 14th european conference, ecsa 2020, proceedings* (pp. 315–332). Springer International Publishing.

Ren, Z., Wang, W., Wu, G., Gao, C., Chen, W., Wei, J., y Huang, T. (2018). Migrating web applications from monolithic structure to microservices architecture. En *Proc. of the 10th asia-pacific symposium on internetware (internetware 2018).* Beijing, China. doi: 10.1145/3275219.3275230

Rochimah, S., y Nuralamsyah, B. (2023). Decomposing monolithic to microservices: keyword extraction and bfs combination method to cluster monolithic's classes. *Jurnal RESTI - Rekayasa Sistem dan Teknologi Informasi*, *7*(2), 169–178.

Taibi, D., y Systä, K. (2019). A decomposition and metric-based evaluation framework for microservices. En *International conference on cloud computing and services science* (pp. 133–149). Springer International Publishing.

Taibi, D., y Systä, K. (2020). A decomposition and metric-based evaluation framework for microservices. En *Communications in computer and information science* (Vol. 1218, pp. 133–149). Springer, Cham. doi: 10.1007/978-3-030-49432-2_7

Trabelsi, I., Abdellatif, M., Abubaker, A., Moha, N., Mosser, S., Ebrahimi-Kahou, S., y Guéhéneuc, Y.-G. (2022). From legacy to microservices: A type-based approach for microservices identification using machine learning and semantic analysis. *Journal of Software: Evolution and Process*, *35*(4), e2503.

Wolfart, D., Assunção, W. K. G., da Silva, I. F., Domingos, D. C. P., Schmeing, E., Donin Villaca, G. L., y do Nascimento Paza, D. (2021). Modernizing legacy systems with microservices: A roadmap. En *Proceedings of the 2021 international conference on software engineering.* doi: 10.1145/3463274.3463334

Würz, H. M., Krämer, M., Kaster, M., y Kuijper, A. (2023). Migrating monolithic applications to function as a service. *Software: Practice Experience*. doi: 10 .1002/spe.3263

Zougari, S., Daoud, M., Abdelouahed, S. M., y Zougari, A. (2024). Automating the recognition of microservices from business process analysis. En *2024 international conference on intelligent systems and computer vision (iscv).* doi: 10.1109/ISCV60512.2024.10620133

A. ANEXOS

A.1 ANEXO 1: INSTRUMENTO DE RECOLECCIÓN DE DATOS

Maecenas non massa. Vestibulum pharetra nulla at lorem. Duis quis quam id lacus dapibus interdum. Nulla lorem. Donec ut ante quis dolor bibendum condimentum. Etiam egestas tortor vitae lacus. Praesent cursus. Mauris bibendum pede at elit. Morbi et felis a lectus interdum facilisis. Sed suscipit gravida turpis. Nulla at lectus. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Praesent nonummy luctus nibh. Proin turpis nunc, congue eu, egestas ut, fringilla at, tellus. In hac habitasse platea dictumst.

A.2 ANEXO 2: VALIDACIÓN DE INSTRUMENTOS

Vivamus eu tellus sed tellus consequat suscipit. Nam orci orci, malesuada id, gravida nec, ultricies vitae, erat. Donec risus turpis, luctus sit amet, interdum quis, porta sed, ipsum. Suspendisse condimentum, tortor at egestas posuere, neque metus tempor orci, et tincidunt urna nunc a purus. Sed facilisis blandit tellus. Nunc risus sem, suscipit nec, eleifend quis, cursus quis, libero. Curabitur et dolor. Sed vitae sem. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Maecenas ante. Duis ullamcorper enim. Donec tristique enim eu leo. Nullam molestie elit eu dolor. Nullam bibendum, turpis vitae tristique gravida, quam sapien tempor lectus, quis pretium tellus purus ac quam. Nulla facilisi.