UNIVERSIDAD PERUANA UNIÓN
FACULTAD DE INGENIERIA Y ARQUITECTURA
Escuela Profesional de Ingeniería de Sistemas

Modelo de Observabilidad para el monitoreo de microservicios en entornos distribuidos.

Por:
Autor 1 (Christian Giovani Supo Condori)
Autor 2 (Gustavo Alberto Salluca Solis)

Asesor:
Dr.

Juliaca, Abril de 2025

Contents

1 Planteamiento del Problema

1.1 Identificación del Problema

En la actualidad, el crecimiento de los sistemas distribuidos y la adopción de arquitecturas basadas en microservicios ha llevado a su implementación tanto por grandes empresas como Amazon, Netflix, Spotify y Twitter, como por un número creciente de pequeñas y medianas empresas. Esto se debe a sus múltiples beneficios, como la capacidad de desarrollar, desplegar y escalar servicios de manera independiente. Entre las ventajas destacadas se incluyen una mayor flexibilidad, facilidad de mantenimiento y una mayor tolerancia a fallos. Esto permite que las organizaciones se adapten más rápidamente a los cambios y continúen operando sin necesidad de detener todo el sistema cuando ocurre un problema [1] [2] [3].

Sin embargo, aunque la arquitectura de microservicios ofrece los beneficios y ventajas anteriormente mencionados, también enfrenta ciertos desafíos debido a su naturaleza distribuida [4]. Uno de los problemas más comunes es la gestión de la comunicación entre servicios, que puede dar lugar a problemas de latencia y sincronización. Además, está el tema de la resiliencia; un fallo en un servicio puede afectar a otros, lo que hace que sea necesario implementar patrones específicos para garantizar una alta disponibilidad [5]. Por último, la complejidad en la observabilidad se presenta como uno de los mayores desafíos, ya que monitorear y correlacionar el comportamiento de múltiples servicios distribuidos se vuelve tanto esencial como complicado [6].

La observabilidad en la arquitectura de microservicios se refiere a la capacidad de monitorear cómo se comporta un sistema distribuido. Esto se logra principalmente mediante registros, métricas y trazas, lo que nos brinda una visión clara de la salud del sistema y nos ayuda a detectar problemas rápidamente. Dado que los microservicios operan de manera distribuida en diferentes instancias, la observabilidad se convierte en un enfoque fundamental para construir sistemas robustos y eficientes. Esto incluye desde el rendimiento hasta la detección y diagnóstico de fallos y anomalías, así como la evaluación del estado general de las aplicaciones. Esta estrategia proactiva permite a los equipos correlacionar eventos entre servicios, identificar problemas y resolverlos de manera ágil, sin afectar al sistema en su conjunto [7] [8].

En este contexto, la observabilidad se convierte en uno de los retos más importantes dentro de la arquitectura de microservicios [9]. Esto es especialmente cierto si lo comparamos con las arquitecturas monolíticas, donde el monitoreo es más sencillo y centralizado. A diferencia de los sistemas monolíticos, los microservicios exigen una visibilidad completa y en tiempo real de todos los componentes distribuidos. Esto significa que es necesario supervisar su rendimiento, analizar los patrones de tráfico y detectar posibles fallos antes de que afecten al sistema en su totalidad [10] [11].

Una visibilidad limitada y la creciente complejidad en la gestión de microservicios pueden comprometer seriamente la fiabilidad y el rendimiento de los sistemas. Sin una solución de observabilidad adecuada, las empresas pueden enfrentar dificultades para identificar problemas

de rendimiento, rastrear errores o realizar diagnósticos rápidos en un entorno distribuido [12]. El monitoreo de microservicios, que incluye el seguimiento de la salud de cada servicio, el análisis de las interacciones entre ellos y la detección temprana de fallos, requiere una visión unificada que permita obtener datos en tiempo real y gestionar estos componentes de manera eficiente [13].

La falta de una solución adecuada de observabilidad en arquitecturas distribuidas puede llevar a fallos operativos costosos, con un impacto negativo en la experiencia del usuario y en la eficiencia de los equipos de desarrollo [14]. Sin una visibilidad completa de cada microservicio y su interacción con otros componentes, las empresas corren el riesgo de pasar por alto problemas críticos hasta que estos afecten de manera significativa el rendimiento del sistema [15].

Existen diversas soluciones consolidadas para el monitoreo de microservicios, entre ellas Prometheus, Jaeger, OpenTelemetry y Grafana, entre otros. Si bien estas herramientas son efectivas y ampliamente adoptadas, su implementación conjunta puede implicar una serie de desafíos. Requieren configuraciones complejas, coordinación entre múltiples componentes, y un entorno que cumpla con ciertos estándares de arquitectura, escalabilidad y flexibilidad. En muchos casos, la falta de experiencia técnica, recursos limitados o restricciones de tiempo dificultan su adopción adecuada. Cuando no se cumplen estos requisitos, es poco probable que se logre una solución de observabilidad efectiva, lo que puede comprometer el rendimiento del sistema y dificultar la detección oportuna de fallos. [3] [7] [16].

1.2 Justificación

La creciente adopción de sistemas distribuidos y arquitecturas basadas en microservicios ha permitido a las organizaciones mejorar en términos de escalabilidad, flexibilidad y resiliencia en el desarrollo de sus aplicaciones [17] [18] [19]. Sin embargo, estas ventajas también han introducido nuevos desafíos, entre ellos, la necesidad de una observabilidad eficiente para gestionar entornos cada vez más complejos y dinámicos [20]

La observabilidad en sistemas distribuidos se vuelve fundamental para garantizar la operatividad, identificar fallos tempranamente y mantener altos niveles de desempeño [21]. Actualmente, si bien existen herramientas como Prometheus, Jaeger, OpenTelemetry y Grafana, estas soluciones suelen ser complejas de configurar, integrar y mantener, especialmente para organizaciones con recursos limitados. Además, la curva de aprendizaje asociada a su implementación representa una barrera importante que puede afectar la adopción generalizada de buenas prácticas de monitoreo [20].

En este contexto, la presente investigación propone el diseño y validación de un modelo de observabilidad que integre los tres pilares fundamentales (logs, métricas y trazas) bajo un enfoque sistemático, adaptable y escalable. Este modelo buscará establecer lineamientos conceptuales y prácticos que permitan guiar la implementación efectiva de observabilidad en entornos distribuidos, reduciendo la complejidad técnica y promoviendo una mejor toma de decisiones.

1.3 Estado del Arte

Una propuesta destacada es la arquitectura MicroCM, presentada por Wang, Tian y Ying , que introduce un sistema de monitoreo en la nube enfocado en la gestión de invocaciones de microservicios. MicroCM permite la recopilación y análisis en tiempo real de datos operacionales, facilitando la identificación de cuellos de botella, la detección de anomalías y la optimización del rendimiento de servicios distribuidos. La incorporación de esta solución en entornos de microservicios mejora significativamente el monitoreo y la estabilidad de las aplicaciones. [22]

Complementando esta línea de trabajo, en se propone un marco ligero y extensible para la telemetría distribuida. Esta solución automatiza la configuración del monitoreo utilizando la especificación OpenAPI (OAS), eliminando la necesidad de intervención manual. Al generar automáticamente las configuraciones, se facilita la integración de la observabilidad en arquitecturas complejas, reduciendo los costos de implementación y mejorando la cobertura del sistema. [20]

Por otro lado, la necesidad de rastrear el flujo de eventos en sistemas basados en Event Sourcing es abordada en . Los autores proponen un enfoque innovador que permite vincular cada solicitud a los eventos registrados, proporcionando una visibilidad completa sobre el comportamiento del sistema. Esta mejora en la observabilidad resulta crítica para facilitar el diagnóstico y la resolución de problemas en arquitecturas distribuidas, donde la trazabilidad de las acciones es esencial. [23]

En un contexto de múltiples proveedores de servicios en la nube, el artículo introduce una arquitectura de monitoreo distribuido para JointCloud Computing. Esta arquitectura define un conjunto de colectores distribuidos capaces de recopilar métricas de rendimiento y estado de los servicios en tiempo real, abordando así los desafíos de heterogeneidad, escalabilidad y resiliencia presentes en entornos dinámicos y heterogéneos. [24]

Finalmente, desde una perspectiva práctica, el estudio presentado en analiza las prácticas de diseño, monitoreo y prueba de sistemas de microservicios a partir de entrevistas con profesionales de la industria. Se destacan patrones comunes como el uso de API Gateway, Backend for Frontend y la adopción de métricas para evaluar el uso de recursos y balanceo de carga. Estos hallazgos son relevantes para comprender los desafíos reales y las estrategias aplicadas en la mejora de la eficiencia y la observabilidad en sistemas distribuidos modernos. [25]

En conjunto, estos trabajos ofrecen una visión integral de las soluciones actuales en monitoreo y observabilidad de sistemas distribuidos, evidenciando avances importantes y áreas de oportunidad para futuras investigaciones.

1.4 Objetivos

1.4.1 Objetivo General

Diseñar un modelo de observabilidad basado en los tres pilares: logs, métricas y trazas, para el monitoreo de microservicios en entornos distribuidos.

1.4.2 Objetivos Específicos

- Diseñar el modelo de observabilidad que permita la captura, procesamiento y correlación de datos de telemetría (métricas, logs y trazas)
- Implementar el modelo de observabilidad propuesto, conforme al diseño previamente definido, minimizando la complejidad de configuración y operación.
- Evaluar el modelo de observabilidad implementado mediante pruebas controladas, utilizando métricas como MTTD, MTTR, cobertura de trazabilidad, correlación entre señales y Golden Signals, con el fin de validar su eficacia en la detección de fallos, el análisis del rendimiento y la estabilidad del sistema.

2 Metodología

2.1 Diseño Metodológico

La investigación será de tipo aplicativa y de nivel experimental. Se busca diseñar, implementar y validar una herramienta de observabilidad para microservicios en arquitecturas distribuidas, con el fin de mejorar el monitoreo de los servicios y sus interacciones. La metodología será de carácter cuantitativo, lo que permitirá realizar mediciones sobre el desempeño de la herramienta desarrollada, su eficiencia, implementación y facilidad de uso.

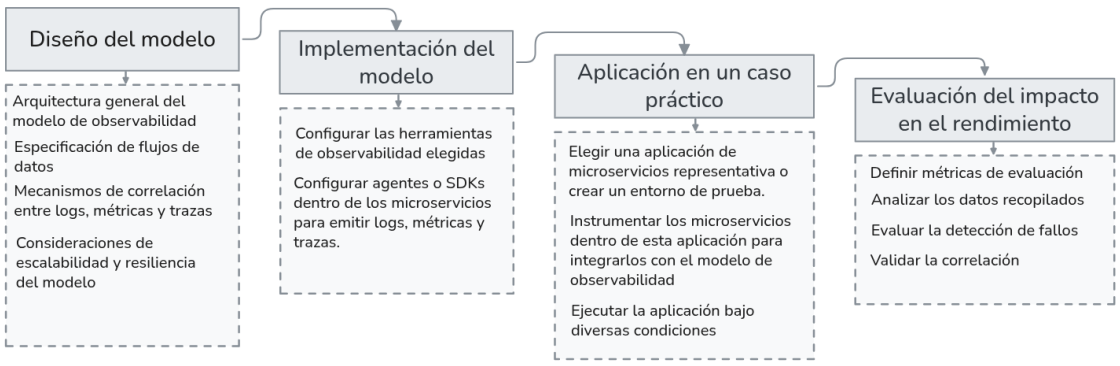

Figure 1: Esquema del diseño metodológico adaptada [20] [22]

2.2 Diseño de la construcción de la propuesta de ingeniería

2.2.1 Diseño del modelo

En esta fase, se crea el plano arquitectónico del modelo de observabilidad, incorporando los conocimientos obtenidos de la revisión del estado del arte.

- **Arquitectura general del modelo de observabilidad:** Esto implica definir la estructura de alto nivel del modelo de observabilidad, describiendo cómo interactuarán los diferentes componentes (por ejemplo, recolectores de datos, procesadores, almacenamiento, herramientas de visualización).

- **Especificación de flujos de datos:** Este paso detalla las rutas que seguirán los logs, las métricas y las trazas dentro del modelo, desde su generación en los microservicios hasta su almacenamiento y presentación final. Esto incluye la definición de formatos de datos, protocolos y pasos de transformación.

- **Mecanismos de correlación entre logs, métricas y trazas:** Un aspecto crítico de la observabilidad es la capacidad de vincular datos de diferentes fuentes. Esto implica diseñar mecanismos que permitan la correlación de logs, métricas y trazas, posibilitando una comprensión holística del comportamiento del sistema y un diagnóstico eficiente de problemas.

- **Consideraciones de escalabilidad y resiliencia del modelo:** Dada la naturaleza dinámica de los microservicios, el modelo debe diseñarse para manejar volúmenes de datos crecientes y complejidades del sistema. Este paso se centra en garantizar que el propio modelo de observabilidad sea escalable y resistente a fallos, evitando que se convierta en un cuello de botella.

2.2.2 Implementación del modelo

Esta fase traduce el modelo diseñado en un sistema en funcionamiento utilizando las herramientas seleccionadas.

- **Configurar las herramientas de observabilidad elegidas:** Las herramientas de observabilidad seleccionadas (por ejemplo, Prometheus, Jaeger, Grafana) se configurarán de acuerdo con la arquitectura y los flujos de datos definidos. Esto implica configurar sus respectivos servicios, bases de datos y canales de comunicación.

- **Configurar agentes o SDKs dentro de los microservicios o crear un entorno de prueba:** Para recopilar datos de telemetría, se integrarán agentes o Kits de Desarrollo de Software (SDKs) en los microservicios. Este paso implica instrumentar los microservicios para emitir los logs, métricas y trazas necesarios en un formato compatible con las herramientas de observabilidad elegidas. Se utilizará una aplicación de microservicios representativa o se creará un entorno de prueba dedicado para este propósito.

2.2.3 Aplicación en un caso práctico

Esta fase implica el despliegue y la prueba del modelo de observabilidad implementado en un entorno de microservicios real o simulado.

- **Elegir una aplicación de microservicios representativa o crear un entorno de prueba:** Se seleccionará una aplicación de microservicios específica, ya sea una existente o un entorno de prueba recién desarrollado, como objeto para aplicar el modelo de observabilidad.

- **Instrumentar los microservicios dentro de esta aplicación para integrarlos con el modelo de observabilidad:** Este paso implica la integración real de los

agentes o SDKs previamente configurados en la aplicación de microservicios elegida, asegurando que los datos de telemetría se generen y se envíen correctamente al sistema de observabilidad.

- **Ejecutar la aplicación bajo diversas condiciones:** Para evaluar a fondo el modelo, la aplicación de microservicios se ejecutará bajo varios escenarios, incluyendo operación normal, condiciones de estrés y fallos simulados, para observar cómo el modelo de observabilidad captura y presenta los datos.

2.2.4 Evaluación del impacto en el rendimiento

La fase final se centra en evaluar la eficacia y eficiencia del modelo de observabilidad implementado.

- **Definir métricas de evaluación:** Se establecerán métricas claras y medibles para cuantificar el rendimiento y la utilidad del modelo de observabilidad. Estas métricas podrían incluir la latencia de la recopilación de datos, la eficiencia del almacenamiento, la precisión de la correlación y el tiempo necesario para detectar y diagnosticar problemas.

- **Analizar los datos recopilados:** Se analizarán los datos de telemetría recopilados durante la fase de *Aplicación en un caso práctico* para comprender el comportamiento de los microservicios y la eficacia del modelo de observabilidad en la captura y presentación de esta información.

- **Evaluar la detección de fallos:** Esto implica evaluar qué tan rápida y precisamente el modelo de observabilidad puede identificar y señalar varios tipos de fallos dentro de la arquitectura de microservicios.

- **Validar la correlación:** Se validará la capacidad del modelo para vincular eficazmente logs, métricas y trazas entre diferentes microservicios, a fin de confirmar su capacidad de proporcionar una vista unificada y completa del comportamiento del sistema distribuido.

2.3 Aspectos Éticos

3 Administración del Proyecto

3.1 Cronograma de Actividades

Descripción de Actividades		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Proyecto	Búsqueda de información	x	x	x																	
	Elaboración del perfil de proyecto		x	x																	
Ejecución	Diseño del modelo				x	x	x														
	Implementación del modelo						x	x	x	x	x	x									
	Aplicación en un caso práctico									x	x	x	x	x							
	Evaluación del impacto en el rendimiento												x	x							
Redacción	Redacción de borradores del artículo					x 01						x 02		x 03	x 04	x					
	Dictaminación del artículo																x	x			
	Revisión y corrección del artículo					x						x	x	x	x	x	x	x			
Cierre	Sumisión del artículo																		x		
	Sustentación																			x	
	Entrega del documento final al repositorio																			x	x

Figure 2: Representación gráfica del cronograma de actividades

3.2 Presupuesto Proyectado

Tipo de Recursos	Cantidad	Precio por Unidad (S/)	Precio Total (S/)
1. Equipos de computación y tecnología			
Laptop o PC de desarrollo	2	4000.00	8000.00
Servidor para pruebas	12 meses	120.00	1440.00
Licencia de sistema operativo	2	250.00	500.00
2. Software y licencias			
Licencia de IDE profesional	2	600.00	1200.00
Librerías y herramientas de observabilidad	-	Gratuito	Gratuito
Backup en la nube (Google Drive 100 GB)	12 meses	8.00	96.00
Software de detección de plagio	1	120.00	120.00
3. Recursos académicos			
Adquisición de artículos científicos	5 artículos	120.00	600.00
Traducción profesional de artículos	1 artículo	80.00	80.00
Publicación de artículo científico en revista indexada	1	1500.00	1500.00
4. Servicios para el proyecto			
Servicio de Internet	12 meses	120.00	1440.00
Asesoría técnica profesional	5 horas	100.00	500.00
Cursos adicionales de certificación	2 cursos	250.00	500.00
5. Costos administrativos de tesis			
Gastos de impresión y empastado de tesis	2 ejemplares	80.00	160.00
Movilidad y transporte	10 viajes	20.00	200.00
Alimentación extra durante el desarrollo	10 sesiones	30.00	300.00
Comunicación móvil	6 meses	30.00	180.00
		Total del Presupuesto	**16.816.00**

Table 1: Presupuesto estimado para el desarrollo del software de observabilidad en arquitecturas de microservicios

3.3 Financiamiento

Table 2: Partición del financiamiento para el desarrollo del proyecto.

Fuente de financiamiento	**Monto (S/.)**	**% de apoyo**
Investigador 1	8.006.4	40%
Investigador 2	8.006.4	40%
Externo	4.003.2	20%
Total	**20,016**	**100%**

4 Referencias Bibliográficas

References

[1] D. Bajaj, U. Bharti, A. Goel, and S. C. Gupta, "A prescriptive model for migration to microservices based on sdlc artifacts," *Journal of Web Engineering*, vol. 20, no. 3, pp. 817–852, 2021.

[2] J. Soldani, J. Khalili, and A. Brogi, "Offline mining of microservice-based architectures (extended version)," *SN Computer Science*, vol. 4, no. 3, p. 304, 2023. [Online]. Available: https://doi.org/10.1007/s42979-023-01721-4

[3] Y. Wang, H. Kadiyala, and J. Rubin, "Promises and challenges of microservices: an exploratory study," *Empirical Software Engineering*, vol. 26, no. 4, p. 63, May 2021. [Online]. Available: https://doi.org/10.1007/s10664-020-09910-y

[4] G. Liu, B. Huang, Z. Liang, M. Qin, H. Zhou, and Z. Li, "Microservices: architecture, container, and challenges," in *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, 2020, pp. 629–635.

[5] N. C. Mendonca, C. M. Aderaldo, J. Camara, and D. Garlan, "Model-based analysis of microservice resiliency patterns," in *2020 IEEE International Conference on Software Architecture (ICSA)*, 2020, pp. 114–124.

[6] P. Jamshidi, C. Pahl, N. C. Mendonça, J. Lewis, and S. Tilkov, "Microservices: The journey so far and challenges ahead," *IEEE Software*, vol. 35, no. 3, pp. 24–35, 2018.

[7] M. C. Borges, J. Bauer, S. Werner, M. Gebauer, and S. Tai, " Informed and Assessable Observability Design Decisions in Cloud-Native Microservice Applications ," in *2024 IEEE 21st International Conference on Software Architecture (ICSA)*. Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2024, pp. 69–78. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICSA59870.2024.00015

[8] U. Faseeha, H. Jamil Syed, F. Samad, S. Zehra, and H. Ahmed, "Observability in microservices: An in-depth exploration of frameworks, challenges, and deployment paradigms," *IEEE Access*, vol. 13, pp. 72 011–72 039, 2025.

[9] A. P. Perumal, "Cloud-native architecture observability and compliance challenges: A comprehensive reference architecture approach," *Library Progress (International)*, vol. 44, pp. 25 718–25 723, 11 2024.

[10] S. Niedermaier, F. Koetter, A. Freymann, and S. Wagner, "On observability and monitoring of distributed systems – an industry interview study," in *Service-Oriented Computing*, S. Yangui, I. B. Rodriguez, K. Drira, and Z. Tari, Eds. Cham: Springer International Publishing, 2019, pp. 36–52.

[11] J. Parmar, S. Sanghavi, V. Prasad, and P. Shah, "Microservice architecture observability tool analysis," in *Soft Computing and Signal Processing*, V. S. Reddy, V. K. Prasad, J. Wang, and K. T. V. Reddy, Eds. Singapore: Springer Nature Singapore, 2023, pp. 1–8.

[12] H. Song, H. Ji, Y. Yu, and B. Xie, "A review of observability issues in hospital information system," in *2022 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 2022, pp. 1–7.

[13] I. Oumoussa and R. Saidi, "Evolution of microservices identification in monolith decomposition: A systematic review," *IEEE Access*, vol. 12, pp. 23 389–23 405, 2024.

[14] D. Berardi, S. Giallorenzo, J. Mauro, A. Melis, F. Montesi, and M. Prandini, "Microservice security: a systematic literature review," *PeerJ Computer Science*, vol. 7, p. e779, 01 2022.

[15] A. Guha, "Model based control for microservices applications," in *2020 IEEE Infrastructure Conference*, 2020, pp. i–i.

[16] L. Giamattei, A. Guerriero, R. Pietrantuono, S. Russo, I. Malavolta, T. Islam, M. Dînga, A. Koziolek, S. Singh, M. Armbruster, J. Gutierrez-Martinez, S. Caro-Alvaro, D. Rodriguez, S. Weber, J. Henss, E. F. Vogelin, and F. S. Panojo, "Monitoring tools for devops and microservices: A systematic grey literature review," *Journal of Systems and Software*, vol. 208, p. 111906, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121223003011

[17] P. Di Francesco, P. Lago, and I. Malavolta, "Architecting with microservices: A systematic mapping study," *Journal of Systems and Software*, vol. 150, pp. 77–97, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121219300019

[18] S. Li, H. Zhang, Z. Jia, C. Zhong, C. Zhang, Z. Shan, J. Shen, and M. A. Babar, "Understanding and addressing quality attributes of microservices architecture: A systematic literature review," *Information and Software Technology*, vol. 131, p. 106449, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584920301993

[19] X. Zhou, S. Li, L. Cao, H. Zhang, Z. Jia, C. Zhong, Z. Shan, and M. A. Babar, "Revisiting the practices and pains of microservice architecture in reality: An industrial

inquiry," *Journal of Systems and Software*, vol. 195, p. 111521, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121222001972

[20] M. Otero, J. M. García, and P. Fernandez, "An extensible lightweight framework for distributed telemetry of microservices," *Sustainable Computing: Informatics and Systems*, vol. 46, p. 101100, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210537925000204

[21] S. Niedermaier, F. Koetter, A. Freymann, and S. Wagner, "On observability and monitoring of distributed systems – an industry interview study," in *Service-Oriented Computing*, S. Yangui, I. Bouassida Rodriguez, K. Drira, and Z. Tari, Eds. Cham: Springer International Publishing, 2019, pp. 36–52.

[22] R. Wang, G. Tian, and S. Ying, "Microcm: A cloud monitoring architecture for microservice invocation," *Computer Networks*, vol. 238, p. 110121, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128623005662

[23] S. Lima, J. Correia, F. Araujo, and J. Cardoso, "Improving observability in event sourcing systems," *Journal of Systems and Software*, vol. 181, p. 111015, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121221001126

[24] Y. Wu, L. Wang, R. Yu, X. Huang, and J. Liu, "A distributed monitoring architecture for jointcloud computing," *Future Generation Computer Systems*, vol. 168, p. 107773, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X25000688

[25] M. Waseem, P. Liang, M. Shahin, A. Di Salle, and G. Márquez, "Design, monitoring, and testing of microservices systems: The practitioners' perspective," *Journal of Systems and Software*, vol. 182, p. 111061, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121221001588