Monitoreo y Alertas de Bases de Datos Oracle: Alternativas y Análisis Comparativo de Soluciones Open Source y Low-Cost.

Autor
Juan Pablo Muñoz Arboleda

Director
Jose Alejandro Franco Calderón

Co-Director
Mauricio Garcés Restrepo

Universidad de Bogotá Jorge Tadeo Lozano
Facultad de Ciencias Naturales e Ingeniería
Especialización en Desarrollo de Bases de Datos

Bogotá - Colombia, mayo de 2025

Índice

Índice de Figuras

Índice de Tablas

Resumen

Este trabajo de grado se centra en la evaluación de Zabbix, Prometheus, Nagios, Oracle Enterprise Manager, entre otros, como soluciones de monitoreo de bajo costo para bases de datos Oracle y su infraestructura de servidores. El objetivo principal es identificar herramientas eficientes que permitan a los Administradores de Bases de Datos (DBAs) implementar un monitoreo proactivo del rendimiento y la salud de sus sistemas Oracle, escalable para cualquier número de instancias.

El análisis técnico abarca la monitorización de métricas críticas como:

Sistema de archivos: Utilización de espacio y rendimiento de E/S.

Memoria: Espacio de intercambio (swap) y uso de RAM.

Backups: Estado y tiempo de ejecución de copias de seguridad.

Almacenamiento Oracle: Espacio utilizado en diskgroups y tablespaces.

CPU: Utilización y carga del procesador.

Se busca implementar un sistema de monitoreo en tiempo real que optimice las operaciones del DBA, especialmente en entornos con restricciones presupuestarias.

Se realizará un análisis comparativo detallado de las herramientas seleccionadas, considerando la funcionalidad (capacidades de monitoreo y alertas), implementación (Facilidad de instalación y configuración), escalabilidad (capacidad de manejo de grandes entornos) y alertas (personalización y eficiencia en la notificación de eventos críticos).

Los resultados de esta investigación proporcionarán a los DBAs una guía práctica para la selección e implementación de soluciones de monitoreo que mejoren la gestión de bases de datos Oracle, reduciendo costos y maximizando la disponibilidad. Se demostrará la viabilidad y efectividad de las alternativas de código abierto para el monitoreo de Oracle, permitiendo la optimización de recursos sin comprometer la calidad del servicio.

Abstract

This thesis focuses on evaluating low-cost monitoring solutions—such as Zabbix, Prometheus, Nagios, and Oracle Enterprise Manager—for Oracle databases and their supporting server infrastructure. The primary goal is to identify efficient tools that enable Database Administrators (DBAs) to implement proactive performance and health monitoring for Oracle systems, scalable across any number of instances.

The technical analysis includes the monitoring of critical metrics such as:

File system: Space usage and I/O performance. Memory: Swap space and RAM utilization. Backups: Status and execution time of backup processes. Oracle storage: Usage in diskgroups and tablespaces. CPU: Processor usage and load.

The aim is to implement a real-time monitoring system that enhances DBA operations, particularly in budget-constrained environments.

A comparative analysis of the selected tools will be conducted, focusing on their monitoring and alerting capabilities, ease of installation and configuration, scalability for large environments, and alert customization and effectiveness in notifying critical events.

The results of this research will provide DBAs with a practical guide for selecting and implementing monitoring solutions that improve Oracle database management while reducing costs and maximizing availability. The study will demonstrate the feasibility and effectiveness of open-source alternatives for Oracle monitoring, enabling resource optimization without compromising service quality.

1. Introducción

Las bases de datos Oracle son reconocidas en el mundo laboral por su robustez, fiabilidad y amplia gama de funcionalidades [1]. Son usadas por empresas de todos los tamaños, desde startups hasta corporaciones multinacionales, que confían en su capacidad para gestionar datos, gracias a su trayectoria y/o compatibilidad con el funcionamiento empresarial de hoy día. Sin embargo, en organizaciones emergentes, con recursos limitados o en aquellas que experimentan un rápido crecimiento, el monitoreo y la gestión de bases de datos Oracle presentan desafíos significativos, lo que exige el uso de soluciones ágiles, confiables y eficaces.[2].

El monitoreo manual de múltiples instancias de bases de datos resulta ineficiente por el tiempo que se puede tardar en identificar alertas y errores, además, este método es susceptible a errores humanos durante los procesos de verifiación, aumentando los riesgos operacionales y de esta manera, afectando la productividad e incluso, la posibilidad de incurrir en la pérdida de datos [3]. En conclusión, la detección tardía de problemas de rendimiento puede generar costes elevados y afectar negativamente la productividad, sin omitir que según el mercado de las empresas, tener una para, puede incurrir en multas y sanciones.

Históricamente, Oracle ha ofrecido herramientas propietarias como lo es el conocido Oracle Enterprise Manager (OEM) para abordar estas necesidades y si bien OEM proporciona una amplia gama de funcionalidades de monitoreo y gestión, su implementación y mantenimiento pueden resultar costosos y complejos, especialmente para organizaciones con presupuestos reducidos o limitados que no pueden optar principalmente por esta opción [4].

En respuesta a estas limitaciones, con el pasar de los años ha habido un creciente interés en soluciones de monitoreo de código abierto y de bajo costo y es acá es donde herramientas como Prometheus, Zabbix, Nagios, etc. ofrecen alternativas viables para el monitoreo de bases de datos Oracle, con la ventaja de ser altamente personalizables y adaptables a las necesidades específicas de cada organización, además de diferenciarse por el costo de aplicación, licencias y soporte[5].

Este trabajo de grado tiene como objetivo explorar y analizar estas soluciones alternativas, con el fin de proporcionar una guía para la selección de la herramienta de monitoreo más adecuada para bases de datos Oracle en entornos con recursos limitados, según necesidades.

2. Descripción del Problema

La administración eficiente de bases de datos Oracle representa un aspecto crítico para la continuidad operativa de muchas organizaciones. No obstante, el monitoreo continuo y en tiempo real del rendimiento y la salud de estos sistemas plantea importantes desafíos técnicos y económicos.

Uno de los principales retos es la complejidad inherente de los entornos Oracle, caracterizados por la coexistencia de múltiples instancias, tablespaces, diskgroups y otros componentes que requieren mecanismos de supervisión robustos, automatizados y permanentemente activos. La identificación y resolución de fallos en estos entornos puede demandar altos niveles de tiempo, conocimiento técnico y recursos.

Otro factor determinante es el elevado costo de las soluciones de monitoreo empresariales, como Oracle Enterprise Manager. Este tipo de herramientas, si bien son potentes, resultan financieramente inviables para muchas organizaciones, especialmente aquellas con presupuestos limitados. Esta situación incrementa el riesgo de interrupciones no detectadas a tiempo, con posibles afectaciones en la productividad y cumplimiento normativo.

Asimismo, existe una necesidad creciente de contar con herramientas que permitan establecer alertas personalizadas y monitoreo proactivo, adaptado a las necesidades específicas de cada organización. Estas soluciones deben ser escalables, flexibles y capaces de ajustarse al crecimiento dinámico de las infraestructuras tecnológicas.

En consecuencia, la ausencia de herramientas de monitoreo con adecuada relación costo-beneficio puede comprometer la disponibilidad, rendimiento y seguridad de las bases de datos Oracle, incrementando el riesgo de pérdidas operacionales y afectando la confianza en los sistemas de gestión de datos.

3. Objetivos

3.1. Objetivo General

Evaluar y comparar soluciones de software libre y bajo costo disponibles para la gestión de alertas y monitoreo de bases de datos Oracle y sus servidores asociados, con el objetivo es proporcionar una guía técnica que facilite la selección de herramientas eficaces, optimizando de esta manera, la administración de estos sistemas en contextos con recursos limitados.

3.2. Objetivos Específicos

- Identificar y analizar herramientas de monitoreo open source y de bajo costo disponibles en el mercado aplicables a bases de datos Oracle.
- Comparar las funcionalidades críticas de dichas herramientas, enfocándose en métricas clave como rendimiento, disponibilidad y capacidad de alerta.
- Formular una matriz comparativa que integre criterios técnicos de selección con base en escalabilidad, facilidad de uso, personalización y eficiencia.
- Evaluar aspectos fundamentales como la facilidad de instalación, mantenimiento y personalización de alertas, con base en escenarios simulados y documentación técnica.

4. Requerimientos

4.1. Requerimientos de Negocio

- La solución debe contribuir a la reducción de costos operativos mediante la detección temprana y resolución automatizada de problemas de rendimiento. Asimismo, debe evitar la necesidad de invertir en soluciones comerciales costosas, y generar datos útiles para apoyar la toma de decisiones orientadas a la optimización del servicio.

4.2. Requerimientos Funcionales

El sistema debe permitir:

- Supervisar métricas críticas como uso de CPU, memoria, espacio en disco, latencia de consultas y tasa de transacciones.
- Verificar de forma continua la disponibilidad de las instancias Oracle.
- Monitorear el espacio disponible en tablespaces y diskgroups, así como la correcta ejecución de los procesos de backups.
- Configurar alertas basadas en umbrales y eventos predefinidos, conforme a los requerimientos operativos establecidos por los administradores.

4.3. Requerimientos de los Usuarios

- Los administradores de bases de datos (DBAs) deben contar con la capacidad de configurar alertas personalizadas, definidas por umbrales críticos específicos. Además, deben disponer de dashboards personalizables adaptados a sus necesidades operativas, así como acceso remoto a los sistemas de monitoreo desde distintos dispositivos, conforme a las políticas internas de seguridad de la organización.

4.4. Requerimientos de Implementación

- La solución debe contar con documentación técnica actualizada que facilite su instalación.
- La configuración inicial debe ser intuitiva y permitir la puesta en marcha operativa en el menor tiempo posible.

4.5. Requerimientos de Calidad

- La herramienta debe ser escalable y capaz de operar eficientemente en entornos con múltiples instancias Oracle. Debe manejar grandes volúmenes de datos sin comprometer el rendimiento y garantizar alta disponibilidad del servicio.

5. Estado del Arte

El monitoreo efectivo de bases de datos Oracle es esencial para garantizar la integridad, el rendimiento y la disponibilidad del entorno empresarial. Con el avance de las tecnologías de la información, se han desarrollado numerosas herramientas y estrategias de monitoreo, que van desde soluciones comerciales avanzadas hasta opciones de código abierto altamente flexibles y adaptables.

Soluciones Comerciales de Vanguardia:

Oracle Enterprise Manager (OEM): OEM proporciona un amplio abanico de capacidades, que incluyen el monitoreo del rendimiento, la gestión proactiva de la disponibilidad, el diagnóstico preciso de problemas y la administración eficiente de la configuración [6].

Su naturaleza integral ofrece una visión profunda y detallada del ecosistema Oracle, aunque su costo puede representar una barrera para organizaciones con presupuestos limitados.

ManageEngine Applications Manager: Esta herramienta se distingue por su capacidad para ofrecer un monitoreo exhaustivo de bases de datos Oracle, así como de otros componentes críticos de la infraestructura de TI. Proporciona información detallada y precisa sobre el rendimiento, la disponibilidad y la utilización de recursos [7].

Esta herramienta permite monitorear una gran variedad de servicios, y otorga una gran capacidad de monitoreo de las bases de datos oracle.

Quest Foglight for Oracle: Esta herramienta permite el monitoreo del rendimiento, y ofrece capacidades avanzadas de diagnóstico y optimización para bases de datos Oracle, lo que permite a los administradores de bases de datos (DBA) identificar y resolver problemas de rendimiento de manera proactiva [3].

Soluciones de Código Abierto de Alto Rendimiento:

Zabbix: Zabbix es una plataforma de monitoreo open source ampliamente adoptada en entornos de TI heterogéneos. De acuerdo con su documentación oficial [8], su arquitectura cliente-servidor, junto con una base de datos relacional y una interfaz web, permite la recolección estructurada de métricas. Ofrece plantillas preconfiguradas para entornos Oracle y un sistema de alertas altamente personalizable. No obstante, su configuración inicial puede ser compleja, y la integración con Oracle requiere el uso de plugins adicionales o configuraciones vía SNMP/ODBC.

Prometheus: Prometheus ha ganado reconocimiento en entornos de contenedores y microservicios, gracias a sus capacidades de monitoreo de series temporales y alertas basadas en reglas [9].

Esta herramienta tiene una gran capacidad de monitoreo de métricas, pero requiere de un conocimiento mas avanzado en comparación a otras herramientas.

Nagios: Nagios se destaca como una herramienta de monitoreo de código abierto altamente

personalizable, capaz de supervisar la disponibilidad de servicios y hosts. Su amplia comunidad de usuarios garantiza un soporte continuo y una amplia gama de complementos [10].

Tendencias Emergentes:

Monitoreo en la Nube: La creciente adopción de bases de datos Oracle en entornos de nube ha impulsado el desarrollo de herramientas de monitoreo especializadas para estos entornos.

Además, Oracle Cloud Infrastructure, posee herramientas para la supervisión del estado del conjunto de Oracle Database.

Inteligencia Artificial (IA) y Aprendizaje Automático (ML): La IA y el ML se están integrando cada vez más en las soluciones de monitoreo, con el objetivo de automatizar el diagnóstico de problemas de rendimiento y predecir posibles fallos.

Monitoreo Proactivo: En la actualidad, se observa una tendencia consolidada hacia el monitoreo proactivo, caracterizado por la anticipación de fallos mediante análisis de tendencias y generación automática de alertas. Este enfoque, potenciado por la incorporación de técnicas de inteligencia artificial y aprendizaje automático, permite detectar patrones anómalos y ejecutar acciones preventivas antes de que se produzcan interrupciones del servicio.

6. Marco Teórico

6.1. Arquitectura de Oracle

El monitoreo de bases de datos es una tarea crítica para garantizar la salud, rendimiento y disponibilidad de los sistemas de información. En Oracle, una instancia de base de datos combina procesos de fondo como lo son DBWn, LGWR, SMON, PMON, etc. con áreas de memoria dedicadas: la SGA (área global del sistema) compartida entre procesos y la PGA (área global del programa) privada de cada proceso [11]. Estos componentes gestionan el almacenamiento en disco para los archivos de datos, tablespaces, redo logs y además también fgestionan la comunicación de red (listeners y sesiones de cliente). El monitoreo debe observar todos estos elementos claves: la utilización de CPU y memoria del servidor, la ocupación de espacio en disco y tablespaces, la actividad de sesiones y bloqueos, así como la latencia de red en las conexiones Oracle [12]. De esta forma se puede detectar cuellos de botella (p.ej. saturación de I/O o memoria insuficiente) antes de que impacten la disponibilidad o el rendimiento.

6.2. Monitoreo Proactivo vs Reactivo

Una estrategia de monitoreo proactivo implica la observación continua de los sistemas para anticipar fallos: se definen umbrales y alarmas que detectan desviaciones o tendencias (uso excesivo de CPU, tablespaces llenos, etc.) antes de que causen interrupciones. De este modo es posible prevenir tiempos de inactividad inesperados y optimizar recursos (p.ej. ajustar configuraciones o agregar capacidad a tiempo). En contraste, el monitoreo reactivo responde a incidentes una vez ocurridos (por ejemplo, cuando un usuario reporta lentitud o se alcanza el 100 por ciento de uso de un recurso), lo que suele implicar mayores tiempos de inactividad y esfuerzos de recuperación. Por ello, las mejores prácticas recomiendan favorecer lo proactivo: uso de bases de datos históricas para análisis de tendencias, alertas automáticas al sobrepasar umbrales definidos y revisión periódica de "signos vitales" del sistema [13]. Implementar procesos de revisión preventiva (capacity planning, depuración de logs, optimización de consultas) complementa al monitoreo pasivo y mejora la salud global de la base.

6.3. Métricas Relevantes en Entornos Oracle

En entornos Oracle es vital definir un conjunto de métricas críticas que reflejen el estado del sistema. Entre las más comunes se incluyen:

- Uso de CPU y memoria del servidor: Muestra el consumo de recursos del host. Un alto uso persistente puede indicar consultas ineficientes o falta de capacidad
- Capacidad y uso de tablespaces y disco: Monitorea el espacio libre en los tablespaces de datos y de recuperación (reducción de datos, redo logs). Superar cierto umbral (p.ej. 85–97 por ciento) debe generar alerta.
- Sesiones y conexiones activas: Controla cuántas sesiones simultáneas hay abiertas. Exceder el límite configurado puede provocar rechazos de conexión o bloqueos de usuarios.

- Rendimiento de E/S de disco: Incluye tasas de lectura/escritura y latencia. En bases de datos grandes, colas de E/S altas son síntoma de cuellos de botella de almacenamiento.
- Actividad de caches y buffers: Porcentaje de aciertos en la cache de búfer de Oracle (buffer cache hit ratio). Un bajo ratio significa muchas lecturas físicas y potencial degradación
- Estado de backups y replicación: Vigilancia de la frecuencia y éxito de respaldos, y el retardo de replicación (Data Guard). Estos datos son esenciales en planes de recuperación ante fallos.
- Eventos de bloqueo y logs de transacciones: Número de bloqueos y longitud de la cola de log de transacciones. Incrementos súbitos pueden indicar problemas de concurrencia o transacciones largas.

6.4. Importancia del Sistema de Alertas y Notificaciones

Un sistema de alertas efectivo es el mecanismo que convierte el monitoreo en acción preventiva. Las alertas son notificaciones automáticas que se disparan al sobrepasar umbrales críticos definidos para cada métrica (por ejemplo, tablespace al 97 por ciento de uso). Oracle Enterprise Manager, por ejemplo, emite alertas proactivas cuando detecta condiciones indeseables (espacio bajo, sesiones suspensas, etc.). Estas alertas pueden entregarse vía correo electrónico, SMS u otros canales, y pueden incluso ejecutar acciones predefinidas (p.ej. ejecutar scripts de limpieza).

La configuración de umbrales debe basarse en las características propias de cada sistema, evitando tanto alarmas demasiado sensibles que generen ruido, como alertas demasiado laxas que lleguen muy tarde. Cuando el umbral crítico se libera, es importante que el sistema también genere una notificación de "limpieza" para indicar que el problema fue resuelto. En general, un modelo mixto donde cada alerta tenga niveles de advertencia (warning) y crítico, y donde se prioricen según impacto, ayuda a atender primero las condiciones más severas. La documentación de Oracle destaca este enfoque de monitoreo proactivo mediante alertas configurables, orientado a mantener la base de datos funcionando dentro de límites seguros.

6.5. Herramientas de Monitoreo

En el mercado existen herramientas comerciales especializadas (Oracle Enterprise Manager –OEM– es la solución nativa de Oracle para entornos Oracle) y herramientas de código abierto o de bajo costo (como Zabbix, Prometheus y Nagios) que, si bien no son específicas de Oracle, pueden adaptarse mediante agentes o plugins. OEM es un producto propietario integrado con la base de datos Oracle, mientras que Zabbix, Prometheus y Nagios son soluciones de monitoreo general ampliamente usadas en TI. A continuación se describen cada una con énfasis en su arquitectura, funcionalidad, ventajas y limitaciones.

6.5.1. Oracle Enterprise Manager(OEM)

Oracle Enterprise Manager (OEM) es la plataforma oficial de Oracle para administración y monitoreo centralizado. Emplea una arquitectura multinivel de tres capas: una consola gráfica web o de escritorio (tier 1), uno o varios Oracle Management Servers (tier 2), y agentes inteligentes (Oracle Agents) instalados en cada servidor administrado (tier 3) [14]. El servidor de gestión intermedio (OMS) orquesta la recopilación de datos y envía tareas a los agentes, almacenando toda la información en una base de datos de repositorio central. OEM permite monitorear exhaustivamente instancias Oracle (sesiones, SGA/PGA, procesos internos, alertas del sistema, etc.), además de otros productos Oracle (aplicaciones, middleware, hardware Oracle).

Ventajas: Su integración nativa con Oracle ofrece métricas granulares (vistas dinámicas vs, AWR, ADDM, alertas predefinidas) y automatización de tareas (p.ej. Job Scheduler, diagnósticos), facilitando el monitoreo de la salud general [15]. Además, OEM permite establecer umbrales de uso para generar alertas automatizadas ante condiciones críticas, como la ocupación excesiva de un tablespace.

Limitaciones: Al ser producto de nivel empresarial, OEM puede ser complejo de instalar y administrar, con requerimientos de licencia adicionales para algunos diagnósticos avanzados. Su interfaz puede resultar pesada y orientada exclusivamente a entornos Oracle, por lo que es menos flexible para infraestructuras heterogéneas. Además, la personalización fuera de las capacidades estándar de Oracle suele ser limitada.

6.5.2. Zabbix

Zabbix es un sistema de monitoreo open-source ampliamente usado para infraestructura TI. Su arquitectura consta de un servidor central que recolecta y procesa métricas, agentes Zabbix instalados en los hosts a monitorear, una base de datos relacional donde se almacena la información recogida, y una interfaz web (frontend) para configuración y visualización [16]. Los agentes pueden operar en modo activo (inician la conexión al servidor) o pasivo (esperan que el servidor solicite datos).

Zabbix permite monitorizar disponibilidad y rendimiento (uso de CPU, memoria, servicios, procesos, espacio en disco, etc.), generando gráficos y reportes históricos. Incorpora plantillas predefinidas para sistemas populares (incluido Oracle) y un motor de alertas con alto nivel de personalización. También soporta Auto Discovery de dispositivos, y notificaciones vía correo, SMS u otros canales.

Ventajas: Al ser gratuito y extensible, Zabbix ofrece una interfaz moderna y paneles flexibles para visualizar métricas. Su modelo basado en base de datos facilita el almacenamiento de datos históricos. Cuenta con una comunidad activa que produce plugins y plantillas.

Limitaciones: La instalación y configuración inicial puede ser compleja (configurar servidor, agente, DB) y consume recursos considerables según la escala. Para monitorizar Oracle se requiere el Zabbix Agent 2 con plugin de Oracle o el uso de SNMP/ODBC, lo que implica

permisos adecuados en la base de datos. Además, Zabbix no está especializado únicamente en Oracle, por lo que puede necesitar ajustes y scripts adicionales para métricas específicas de Oracle.limitada.

6.5.3. Nagios

Nagios (Core/NXI) es una herramienta de monitoreo veterana, extensible mediante plugins. Su arquitectura básica es un servidor Nagios que ejecuta el demonio principal (scheduler) y coordina chequeos, junto con plugins/agentes (como NRPE o NSClient++) que realizan pruebas locales en cada máquina monitoreada [17]. En esencia, Nagios sigue un modelo host-agent: el servidor central consulta periódicamente los plugins de cada host para verificar servicios, recursos y responde con notificaciones si detecta fallos.

Nagios es capaz de supervisar cualquier recurso para el cual exista un plugin (estado de procesos, servicios de red, espacio en disco, logs, etc.). Incluye un sistema de alertas básico (correo, SMS con add-ons) y permite definir escalados de notificación. Nagios XI (versión empresarial) agrega interfaz web gráfica, informes y soporte oficial.

Ventajas: Es muy configurable y soporta un amplio repertorio de plugins (muchos desarrollados por la comunidad). Su filosofia minimalista permite máxima flexibilidad. Además es gratuito (Core) y cuenta con extensa documentación y comunidad.

Limitaciones: Nagios Core requiere configuración manual basada en archivos de texto, lo que puede ser engorroso. Su interfaz web original es rudimentaria y el escalado puede demandar múltiples instancias (un solo servidor Nagios puede ser un punto de falla bajo alta carga). Asimismo, carece de almacenamiento de métricas a largo plazo fuera de bases externas, y las gráficas no son parte del núcleo (requiere plugins adicionales). En resumen, aunque “Nagios es altamente personalizable”, también es “difícil de instalar, poco intuitivo para nuevos usuarios y puede implicar sobrecarga en los sistemas monitoreados” [18].

A continuación se expone un cuadro comparativo con los elementos más importantes de cada herramienta.

ID	Fase/Actividad	Duración (Semanas)	Fecha de Inicio (Estimada)	Fecha de Fin (Estimada)	Predecesora
1	Fase de Investigación y Recopilación	4	2025-06-02	2025-06-27	
1.1	Identificación de herramientas	2	2025-06-02	2025-06-13	
1.2	Recopilación de documentación técnica	2	2025-06-09	2025-06-20	1.1
1.3	Análisis de arquitecturas Oracle	1	2025-06-16	2025-06-20	1.1
1.4	Estudio de métricas y sistemas de alertas	1	2025-06-23	2025-06-27	1.2, 1.3
2	Fase de Diseño y Configuración	5	2025-06-30	2025-08-01	
2.1	Diseño de la arquitectura del entorno	1	2025-06-30	2025-07-04	1.4
2.2	Instalación y configuración de instancias Oracle	2	2025-07-07	2025-07-18	2.1
2.3	Instalación y configuración de herramientas	3	2025-07-07	2025-07-25	2.1
2.4	Desarrollo de scripts personalizados	2	2025-07-21	2025-08-01	2.2, 2.3
2.5	Configuración de Oracle Exporter	1	2025-07-28	2025-08-01	2.4
3	Fase de Ejecución de Pruebas	4	2025-08-04	2025-08-29	
3.1	Simulación de cargas de trabajo	2	2025-08-04	2025-08-15	2.5
3.2	Recolección de datos de monitoreo	2	2025-08-11	2025-08-22	3.1
3.3	Evaluación de detección y alertas	1	2025-08-18	2025-08-22	3.2
3.4	Pruebas de escalabilidad	1	2025-08-25	2025-08-29	3.3
4	Fase de Análisis Comparativo	3	2025-09-01	2025-09-19	
4.1	Análisis de funcionalidades críticas	1	2025-09-01	2025-09-05	3.4
4.2	Evaluación de instalación y personalización	1	2025-09-08	2025-09-12	4.1
4.3	Comparación de eficiencia y escalabilidad	1	2025-09-08	2025-09-12	4.1
4.4	Integración de criterios en matriz comparativa	0.5	2025-09-15	2025-09-17	4.2, 4.3
4.5	Identificación de fortalezas y debilidades	0.5	2025-09-17	2025-09-19	4.4

7. Solución Propuesta

7.1. Descripción general de la solución

Ante la necesidad de contar con herramientas eficientes, económicas y funcionales para el monitoreo y alertas de bases de datos Oracle, la propuesta de solución consiste en un sistema híbrido que integra herramientas de código abierto como Zabbix, Prometheus y Grafana, complementadas con scripts personalizados para la recolección de métricas específicas de las bases de datos Oracle. El objetivo principal es establecer un monitoreo continuo que minimice el impacto sobre los recursos de procesamiento de las instancias Oracle y permita la visualización y análisis de datos en tiempo real.

La arquitectura del sistema se fundamenta en un flujo secuencial de datos: las métricas son extraídas de las instancias Oracle mediante scripts o exportadores personalizados; posteriormente, son recogidas y almacenadas por Prometheus; luego, Grafana se encarga de la visualización y análisis de los datos; finalmente, Zabbix monitorea la infraestructura y gestiona alertas adicionales.

A continuación, se presenta un esquema simplificado que ilustra el funcionamiento del sistema:

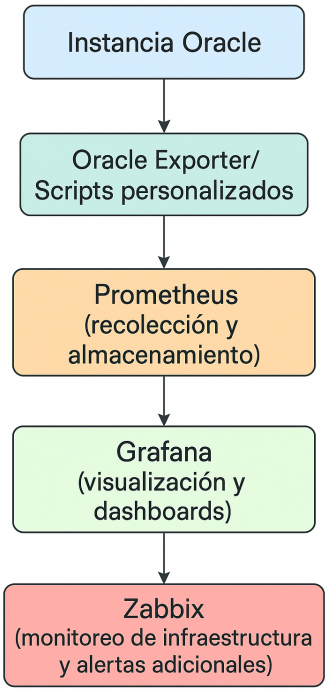

Figura 1: Entorno de pruebas monitoreo

7.2. Selección de Herramientas

Esta solución busca:

- Establecer monitoreo continuo con recolección periódica de métricas clave mediante consultas programadas de los recursos críticos (uso de CPU, I/O, espacio en disco, conexiones activas, entre otros).
- Implementar alertas proactivas ante eventos como caídas de servicio, lentitud o picos de uso.
- Generar reportes históricos para análisis y toma de decisiones.
- Ser escalable, con mínima carga sobre los recursos del servidor Oracle.
- Garantizar trazabilidad y personalización de los indicadores de desempeño (KPIs).

La arquitectura de la solución estará compuesta por:

Herramienta	**Rol Principal**	**Se Interconecta con...**	**Valor Agregado**
Oracle Exporter	Recolección de métricas específicas de Oracle	Bases de Datos Oracle Prometheus	Agente ligero y dedicado que traduce métricas DB a un formato estándar (Prometheus).
Prometheus	Almacenamiento de series temporales (TSDB) y Motor de Alertas	Oracle Exporter Grafana Alertmanager	Eficiente para métricas de series de tiempo; motor de reglas robusto para alertas.
Grafana	Visualización de datos mediante dashboards	Prometheus	Alta flexibilidad para la creación de visualizaciones personalizadas y dinámicas.
Zabbix	Monitoreo de infraestructura y alertamiento granular	Alertmanager Scripts personalizados	Visión completa de la infraestructura (SO, red, hardware); capacidades de alerta extendidas.
Scripts Bash / Python	Integración, automatización de tareas y validaciones adicionales	Oracle Prometheus Zabbix Sistema Operativo	Personalización de la solución y automatización de procesos específicos del entorno empresarial.

Tabla 1: Composición arquitecturas solución

7.3. Definición de Criterios de Evaluación

Para la comparación de las herramientas, se establecieron los siguientes criterios clave, ponderados según su relevancia para el monitoreo de Oracle:

- Funcionalidad: Cobertura de métricas clave (servidor y BD Oracle), capacidad de visualización (dashboards), personalización de monitores.
- Sistema de Alertas: Personalización y eficiencia en la notificación de eventos críticos.
- Implementación: Facilidad de instalación y configuración inicial, calidad de la documentación.
- Escalabilidad: Capacidad para manejar múltiples instancias Oracle, eficiencia en el uso de recursos.
- Costo: Licenciamiento (énfasis en open source/gratuito), soporte, infraestructura requerida.
- Comunidad y Soporte: Disponibilidad de soporte comunitario, actualizaciones, plugins/extensiones.

A modo de ejemplo, se presenta una matriz de evaluación preliminar:

Criterio	Zabbix	Prometheus	Grafana	OEM
Funcionalidad	4	5	4*	5
Alertas	5	5	4*	5
Implementación	4	3	4	2
Escalabilidad	4	5	5	5
Costo	5	5	5	1
Soporte/Comunidad	5	5	5	5

Tabla 2: Matriz evaluación herramientas

7.4. Proceso de implementación

Para establecer un proceso correcto de implmentación, se determina el siguiente conjunto de pasos:

- Diseño del Entorno de Pruebas: Configurar un ambiente controlado con una o varias instancias de bases de datos Oracle y los servidores correspondientes. Instalar y configurar cada una de las herramientas de monitoreo seleccionadas para supervisar este entorno.
- Ejecución de Pruebas: Operar el entorno simulando cargas de trabajo y eventos típicos (alto uso de CPU, llenado de tablespaces, errores, etc.). Recolectar datos de monitoreo y evaluar la respuesta de cada herramienta (detección, alertas, visualización).
- Pruebas de Escalabilidad: Simulación de crecimiento en el número de instancias.
- Evaluación de Alertas y Dashboards: Personalización y pruebas de notificaciones.

- Análisis Comparativo: Utilizar la matriz de criterios para evaluar objetivamente cada herramienta basándose en los resultados de las pruebas y la investigación documental. Identificar fortalezas y debilidades de cada solución en el contexto del monitoreo Oracle.
- Elaboración de Guía Práctica: Sintetizar los hallazgos en una guía que presente los resultados comparativos y ofrezca recomendaciones claras para que los DBAs puedan seleccionar la herramienta más adecuada según sus necesidades específicas.

Se usará una base de datos de prueba, donde Prometheus extraerá datos a través de Scripts que se plasmarán en Grafana. A su vez de la instancia también se extraerá información de Oracle Exporter.

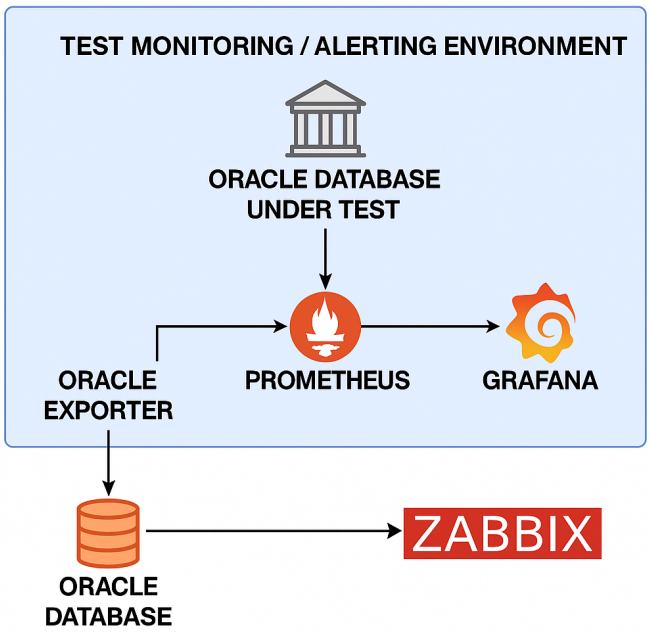

Figura 2: Entorno de pruebas monitoreo

7.5. Proceso de visualización y análisis:

Se realizan los bocetos de las alertas y gráficas esperadas tras configurado el proceso de pruebas usando Grafana.

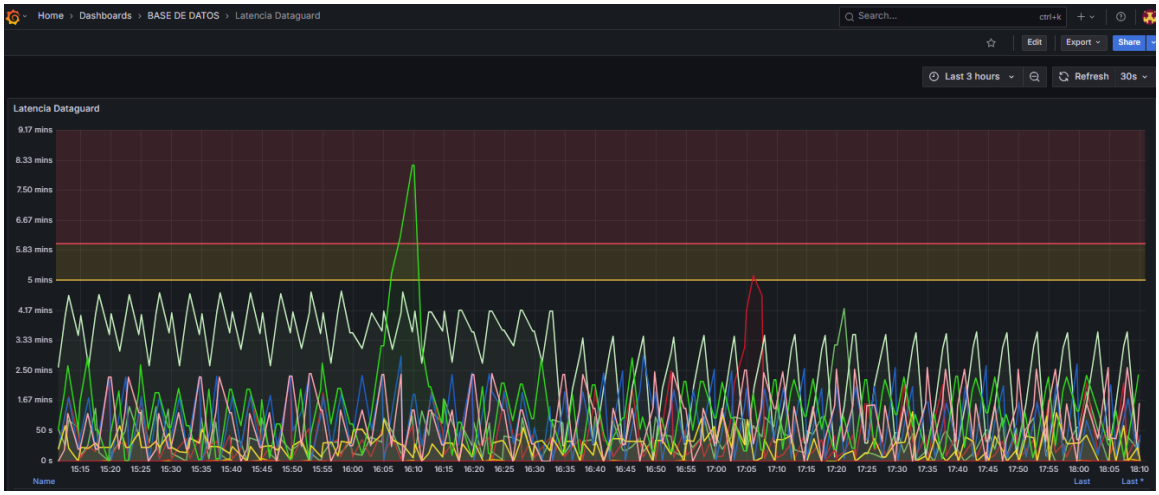

Figura 3: Ejemplo boceto métrica Data Guard

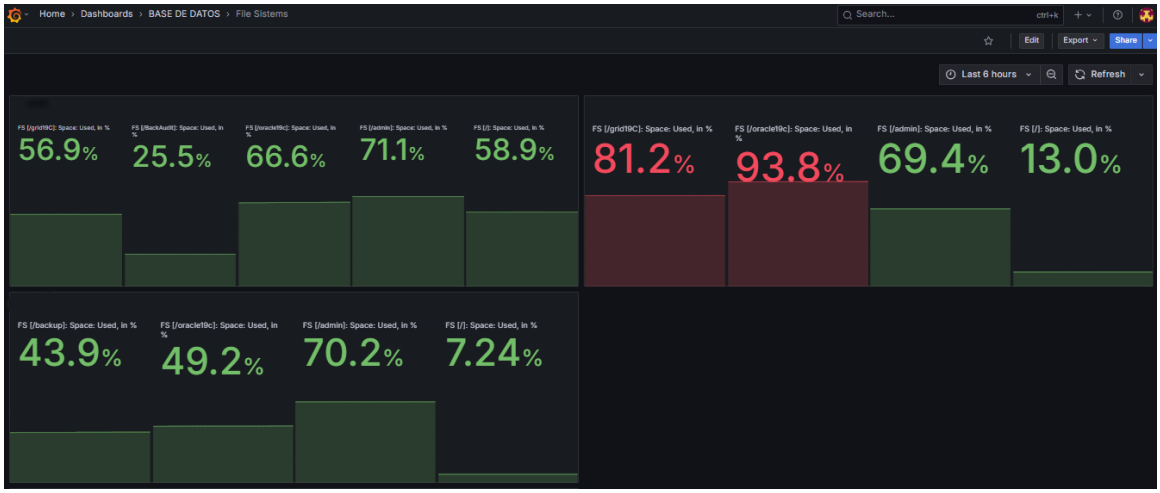

Figura 4: Ejemplo boceto Espacio File Systems

Figura 5: Ejemplo boceto gráfica uso de CPU

CC: NOTIFICACIONES

Host=
Target type=**Database Instance**
Target name
Categories=**Availability**
Message=**The value of Diferencia for is 5**
Severity=**Warning**
Event reported time=**May 28, 2025 2:29:56 PM COT**
Operating System=**AIX**
Platform=**powerpc**
Event Type=**Metric Alert**
Event name=**ME$DGUARD_SYNC_LOGGAP:DIF_LOG_DGUARD**
Metric Group=**ME$DGUARD_SYNC_LOGGAP**
Metric=**DIF_LOG_DGUARD**
Metric value=**5**
Key Value=

Figura 6: Ejemplo correo alerta Data Guard con retraso

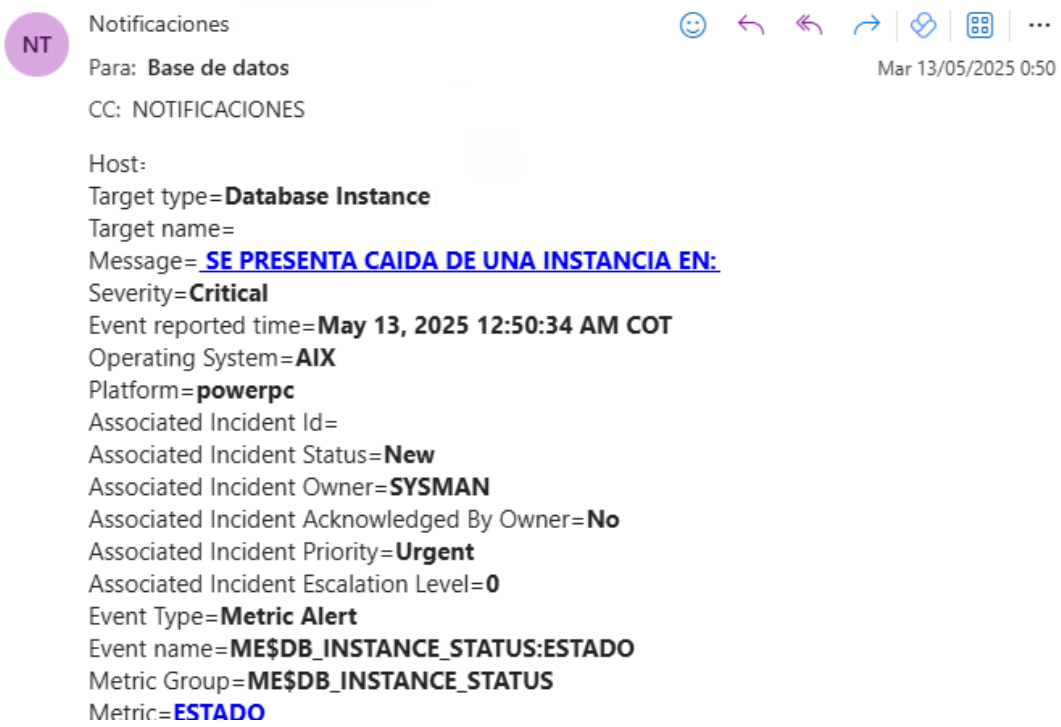

Figura 7: Ejemplo correo alerta instancia caída

8. Planeación del Trabajo

8.1. Descomposición de actividades WBS

A continuación se muestra el Work Breakdown Structure del proyecto.

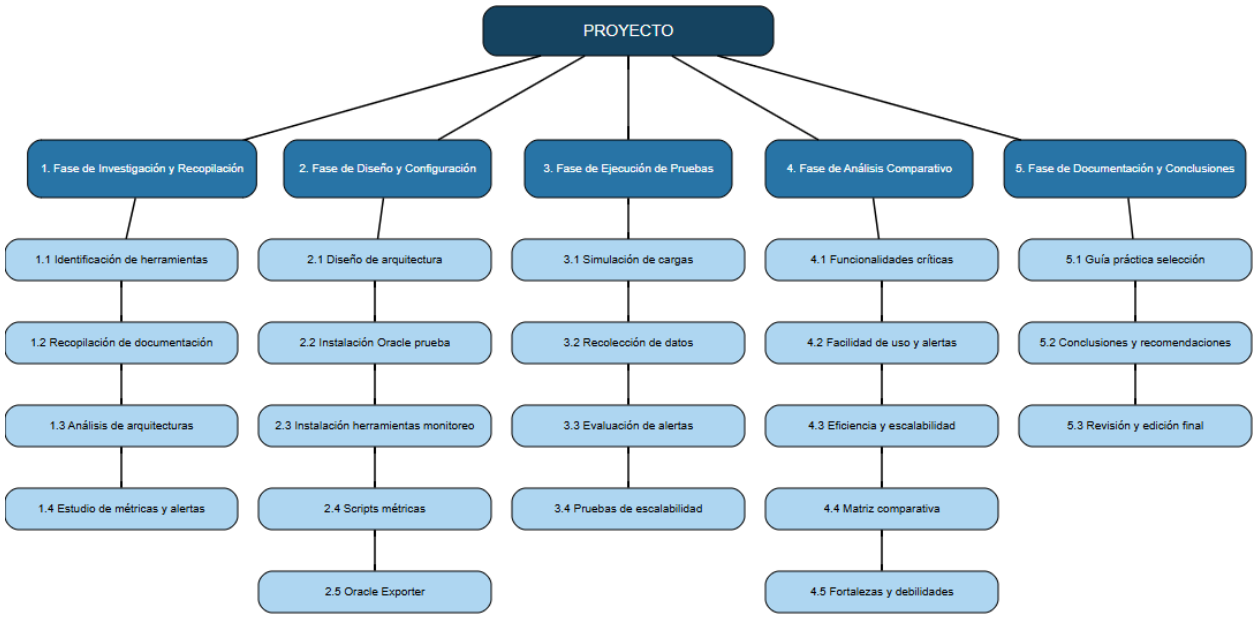

Figura 8: Work Breakdown Structure

8.2. Tabla de actividades

Esta tabla de actividades organiza las actividades desde la definición del problema hasta la consecusión de los objetivos propuestos. Su estructura asegura un flujo que abordará cada temática desde el planteamiento del problema, pasando por la selección de las herramientas a usar, hasta la realización de pruebas hasta la consecusión de los resultados esperados.

ID	Fase/Actividad	Duración (Semanas)	Fecha de Inicio (Estimada)	Fecha de Fin (Estimada)	Predecesora
1	Fase de Investigación y Recopilación	4	2025-06-02	2025-06-27	
1.1	Identificación de herramientas	2	2025-06-02	2025-06-13	
1.2	Recopilación de documentación técnica	2	2025-06-09	2025-06-20	1.1
1.3	Análisis de arquitecturas Oracle	1	2025-06-16	2025-06-20	1.1
1.4	Estudio de métricas y sistemas de alertas	1	2025-06-23	2025-06-27	1.2, 1.3
2	Fase de Diseño y Configuración	5	2025-06-30	2025-08-01	
2.1	Diseño de la arquitectura del entorno	1	2025-06-30	2025-07-04	1.4
2.2	Instalación y configuración de instancias Oracle	2	2025-07-07	2025-07-18	2.1
2.3	Instalación y configuración de herramientas	3	2025-07-07	2025-07-25	2.1
2.4	Desarrollo de scripts personalizados	2	2025-07-21	2025-08-01	2.2, 2.3
2.5	Configuración de Oracle Exporter	1	2025-07-28	2025-08-01	2.4
3	Fase de Ejecución de Pruebas	4	2025-08-04	2025-08-29	
3.1	Simulación de cargas de trabajo	2	2025-08-04	2025-08-15	2.5
3.2	Recolección de datos de monitoreo	2	2025-08-11	2025-08-22	3.1
3.3	Evaluación de detección y alertas	1	2025-08-18	2025-08-22	3.2
3.4	Pruebas de escalabilidad	1	2025-08-25	2025-08-29	3.3
4	Fase de Análisis Comparativo	3	2025-09-01	2025-09-19	
4.1	Análisis de funcionalidades críticas	1	2025-09-01	2025-09-05	3.4
4.2	Evaluación de instalación y personalización	1	2025-09-08	2025-09-12	4.1
4.3	Comparación de eficiencia y escalabilidad	1	2025-09-08	2025-09-12	4.1
4.4	Integración de criterios en matriz comparativa	0.5	2025-09-15	2025-09-17	4.2, 4.3
4.5	Identificación de fortalezas y debilidades	0.5	2025-09-17	2025-09-19	4.4

ID	Fase/Actividad	Duración (Semanas)	Fecha de Inicio (Estimada)	Fecha de Fin (Estimada)	Predecesora
5	Fase de Documentación y Conclusiones	3	2025-09-22	2025-10-10	
5.1	Elaboración de guía práctica	2	2025-09-22	2025-10-03	4.5
5.2	Redacción de conclusiones y recomendaciones	1	2025-09-29	2025-10-03	5.1
5.3	Revisión y edición del documento final	1	2025-10-06	2025-10-10	5.2

Tabla 3: Tabla de actividades

8.3. Diagrama de Gantt

Diagrama de distribución de tiempo previsto para cada una de las actividades planteadas.

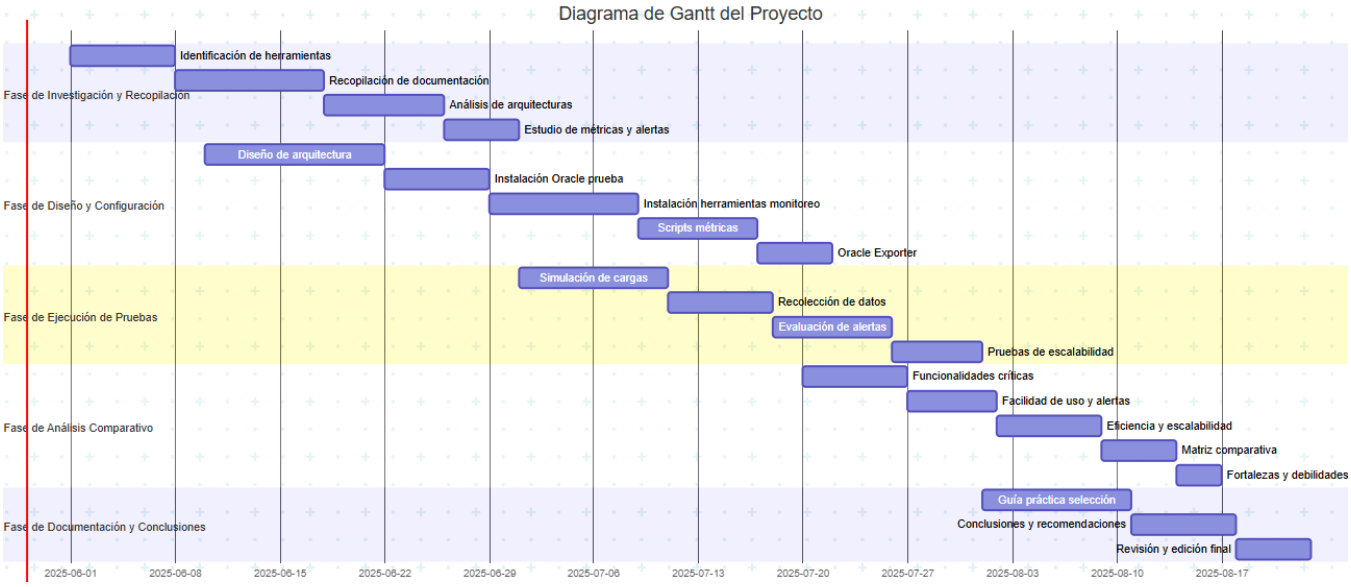

Figura 9: Diagrama de Gantt

9. Presupuesto

Se presenta el presupuesto estimado para llevar a cabo el proyecto, incluyendo costo de personal, de equipos informáticos, de software, licenciamientos por el término de aproximadamente, 2 meses y medio.

Es importante resaltar que este es el costo de investigación y desarrollo de automatización en un entorno de pruebas.

Ítem	Descripción	Costo Individual	Cantidad	Costo Total (COP)
Investigador Principal	Tiempo dedicado a la investigación, diseño, implementación de pruebas, análisis y documentación. (2 meses)	4000000	2 meses	8000000
Desarrollador automatización (ingeniero en sistemas /telecomunicaciones)	Asesoramiento, revisión y guía metodológica.(2 meses)	5000000	2 meses	10000000
Tester	Asesoramiento técnico especializado, revisión y guía. (2 meses)	4500000	2 meses	9000000
Servidor Virtual (VM) para Oracle	Instancia en la nube (AWS EC2, Google Cloud, Azure VM) o local para base de datos Oracle de prueba.	200000	2 meses	400000
Servidor Virtual (VM) para Monitoreo	Instancia para las herramientas de monitoreo (Zabbix, Prometheus, Grafana).	150000	2 meses	300000
Almacenamiento Adicional (SSD)	Para bases de datos y métricas, si la VM base no es suficiente.	50000	2 meses	100000
Base de Datos Oracle (Edición Express/ Standard)	Licencia para uso de prueba o desarrollo (XE es gratuita, Standard tiene costo bajo). Si es gratuita: $0.	0	2 mes	0
Herramientas de Monitoreo (Open Source)	Zabbix, Prometheus, Grafana (gratuitas).	0	2 meses	0
Sistema Operativo (Linux)	Ubuntu Server, CentOS (gratuitos).	0	2 meses	0
Papelería y material de oficina	Hojas, bolígrafos, cuadernos para toma de notas y documentación.	50000	1 unidad	50000
USB/Disco Externo	Para copias de seguridad de configuraciones y datos.	80000	1 unidad	80000
Transporte Local	Desplazamientos a la universidad, reuniones, etc.	30000	8 días	240000
Refrigerios/Almuerzos (sesiones largas)	Si hay reuniones o sesiones de trabajo prolongadas.	15000	10 días	150000
TOTAL PROYECTO				28320000

Tabla 4: Presupuesto del Proyecto

10. Conclusiones

El análisis realizado a lo largo de este proyecto permitió evidenciar que existen alternativas de monitoreo basadas en software libre y de bajo costo que son plenamente viables para entornos con bases de datos Oracle, especialmente cuando se operan bajo restricciones presupuestarias.

En particular, herramientas como Zabbix, Prometheus y Grafana demostraron ser técnicamente competentes para abordar los desafíos del monitoreo integral en sistemas Oracle. A través de una arquitectura híbrida diseñada para este trabajo, se logró establecer un sistema capaz de recolectar, procesar y visualizar en tiempo real métricas esenciales como uso de CPU, memoria, espacio en disco, latencia, disponibilidad de instancias, y estado de procesos críticos como backups y replicación.

Una de las principales fortalezas de las soluciones evaluadas es su capacidad de adaptación a entornos escalables y heterogéneos, permitiendo su implementación en infraestructuras de distintos tamaños. Además, la posibilidad de automatizar tareas mediante scripts personalizados facilita la extensión funcional del sistema de monitoreo y permite una integración más ajustada a las necesidades del entorno empresarial.

El enfoque de monitoreo proactivo propuesto, sustentado en umbrales definidos y alertas configurables, representa un avance significativo frente a esquemas reactivos. Esta capacidad de anticiparse a eventos críticos, en lugar de responder a ellos cuando ya han impactado el sistema, mejora los tiempos de respuesta y optimiza el uso de los recursos técnicos y humanos.

Por otro lado, la comparación detallada entre herramientas permitió generar una guía técnica orientada a la toma de decisiones informadas, útil para administradores de bases de datos que buscan implementar soluciones efectivas sin incurrir en los altos costos de plataformas propietarias como Oracle Enterprise Manager.

Finalmente, se concluye que la adopción de soluciones open source no solo representa un alivio presupuestal, sino que promueve la autonomía tecnológica, incentiva la innovación y fortalece la capacidad operativa de los equipos de gestión de bases de datos.

Referencias

[1] Oracle. «Introduction to Oracle Database.» (), dirección: https://docs.oracle.com/en/database/oracle/oracle-database/21/cncpt/introduction-to-oracle-database.html (visitado 05-04-2025).

[2] M. Engine, «Monitoreo de bases de datos para principiantes: 6 pasos para empezar,» *Manage Engine*, 2024.

[3] *Herramienta de monitoreo y ajuste de rendimiento de Oracle | Foglight*, Quest Software.

[4] *Monitoreo de Oracle – Utilice PRTG para resolver problemas de bases de datos.*

[5] 5icorp, «Monitoreo de bases de datos: 6 pasos para comenzar,» *5icorp*, 2023.

[6] Oracle. «Monitoring Overview.» (), dirección: https://docs.oracle.com/en/enterprise-manager/cloud-control/enterprise-manager-cloud-control/24.1/emmon/monitoring-overview.html (visitado 05-04-2025).

[7] ManageEngine. «Monitoreo de bases de datos Oracle.» (), dirección: https://www.manageengine.com/latam/applications_manager/monitoreo-de-base-de-datos-oracle.html (visitado 05-04-2025).

[8] Zabbix. «Acerca de Zabbix.» (), dirección: https://www.zabbix.com/documentation/current/es/manual/introduction/about (visitado 05-04-2025).

[9] Prometheus. «Overview.» (), dirección: https://prometheus.io/docs/introduction/overview/ (visitado 05-04-2025).

[10] Nagios. «Nagios Documentation.» (), dirección: https://www.nagios.org/documentation/ (visitado 05-04-2025).

[11] Oracle. «Memory Architecture.» (), dirección: https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/memory-architecture.html (visitado 14-05-2025).

[12] ManageEngine. «¿Qué es el Monitoreo de Base de Datos?» (), dirección: https://www.manageengine.com/latam/applications_manager/tech-topics/que-es-el-monitoreo-de-base-de-datos.html (visitado 14-05-2025).

[13] Adivi. «Proactive Monitoring vs. Reactive Monitoring: What's the Difference?» (), dirección: https://adivi.com/blog/proactive-monitoring-vs-reactive-monitoring/ (visitado 14-05-2025).

[14] Oracle. «Oracle Enterprise Manager Architecture.» (), dirección: https://docs.oracle.com/cd/A97335_02/manage.102/a85250/ch1.htm (visitado 14-05-2025).

[15] Oracle. «Configuring Thresholds and Alerts.» (), dirección: https://docs.oracle.com/cd/B14117_01/server.101/b10742/montune.htm (visitado 14-05-2025).

[16] Hawatel. «Introduction to Zabbix Architecture and Key Features.» (), dirección: https://hawatel.com/en/blog/introduction-to-zabbix-architecture-and-key-features/ (visitado 14-05-2025).

[17] B. Stack. «Nagios vs. Zabbix vs. Prometheus: Which One Should You Choose?» (), dirección: https://betterstack.com/community/comparisons/nagios-vs-zabbix-vs-prometheus/ (visitado 14-05-2025).

[18] GeeksforGeeks. «What is Nagios?» (), dirección: https://www.geeksforgeeks.org/what-is-nagios/ (visitado 14-05-2025).

[19] Quest. «Monitoreo del rendimiento de base de datos.» (), dirección: https://www.quest.com/mx-es/solutions/database-performance-monitoring/ (visitado 05-04-2025).

[20] P. FMS. «Monitorización Prometheus: Qué es y cómo funciona.» (), dirección: https://pandorafms.com/blog/es/prometheus-monitoring/ (visitado 14-05-2025).