WHITEPAPER

# CONSIGA UNA ARQUITECTURA DE MICROSERVICIOS EXITOSA

## VENTAJAS

- Desarrollo y mantenimiento más sencillos
- Implementación más rápida e independiente
- Escalabilidad indepen-diente de los componentes de aplicación
- Libertad del compromiso con la tecnología a largo plazo

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

es.redhat.com

## RESUMEN EJECUTIVO

La arquitectura de microservicios es un nuevo estilo arquitectónico para crear servicios de bajo acoplamiento pero autónomos. Las nuevas tendencias en tecnología, como DevOps, la plataforma como servicio (PaaS), los contenedores, así como los métodos de integración y distribución continuas (CI/CD), permiten a las organizaciones crear y gestionar estos sistemas modulares a una escala sin precedentes que supera los enfoques anteriores, como la arquitectura orientada al servicio (SOA). Sin embargo, las organizaciones que refactorizan las aplicaciones monolíticas en microservicios experimentan una gran diversidad de grados de éxito. La clave para utilizar los microservicios con eficacia es disponer de unos conocimientos amplios acerca de cómo y por qué las organizaciones deben usar los microservicios para la creación de aplicaciones.

## MEJORA DE LA ARQUITECTURA ORIENTADA AL SERVICIO

La arquitectura orientada al servicio (SOA) se suele describir como aquellos componentes de aplicación que se comunican para proporcionar servicios a otros componentes a través de una red. El objetivo de la SOA era crear aplicaciones distribuidas resistentes sin complejos componentes centralizados.

Sin embargo, la SOA se acoplaba estrechamente a las estructuras organizativas y se aplicaba como soporte de las nuevas estructuras internas. Por lo tanto, su éxito dependía en gran medida de las capacidades organizativas reestructuradas, resultantes y en la estructura de los equipos que diseñaban la arquitectura. En lugar de crear sistemas de bajo acoplamiento pero autónomos, la SOA creaba sistemas muy frágiles que precisaban de una infraestructura compleja. Además, las primeras implementaciones de SOA creaban dependencia de proveedor, puesto que el middleware propietario a menudo se basaba en una lógica, una persistencia, una gobernanza y una administración centralizadas.

Las arquitecturas de microservicios están empezando a cumplir con las expectativas que prometía la SOA en todos los pasos del proceso de creación de aplicaciones, desde el desarrollo hasta la implementación, pasando por las operaciones. La arquitectura de microservicios se centra en la simplificación de la tecnología para crear sistemas complejos con componentes optimizados. La lógica y la infraestructura de integración centralizadas (basadas en plataformas pesadas no estandarizadas) se sustituyen por la comunicación a través de canalizaciones sencillas y estandarizadas, basadas en HTTP o protocolos de mensajería asíncronos. SOAP, XML y otros formatos de datos y protocolos pesados se sustituyen por JSON ligero a través de REST basado en HTTP. Cada microservicio dispone de su propio almacén de datos, de modo que la gestión y la persistencia centralizadas no son necesarias.

Los microservicios se sirven de las metodologías y prácticas de integración continua (CI) y distribución continua (CD), así como de varios componentes esenciales que no eran tan comunes en la SOA, tales como:

- Programación y persistencia políglotas.
- Contenedores o máquinas virtuales (VM) inmutables.
- Infraestructura como servicio (IaaS) y PaaS elásticas y programables.

### ¿ESTÁ PREPARADO PARA UTILIZAR LOS MICROSERVICIOS?

Pregúntese si su organización ha hecho lo siguiente:

- ¿Ha creado una aplicación monolítica bien estructurada?
- ¿Ha determinado qué necesidades cubrirán los microservicios?
- ¿Ha realineado los equipos en torno a los microservicios?
- ¿Ha adoptado mejores prácticas en DevOps y CI/CD?
- ¿Ha identificado límites de negocio dentro de la aplicación?
- ¿Ha implementado procesos y herramientas para la gestión y orquestación de los microservicios?

## SOLUCIÓN INNOVADORA PARA UNA TI FLEXIBLE Y RECEPTIVA

### IMPLEMENTACIÓN MÁS RÁPIDA

Los microservicios tienen un alcance menor, que viene determinado por un enfoque centrado en los límites de dominio y un modelo de dominio coherente; además requieren menos código. Las estrategias de implementación, que incluyen archivos centrados, autónomos y CI/CD (a menudo empaquetados como contenedores de Linux) permiten implementaciones más rápidas y fiables. Como resultado, el ciclo de vida de desarrollo de software se acelera en términos generales. Las funciones y correcciones de errores nuevas, así como los parches de seguridad totalmente probados, también se publican con mayor rapidez.

### CONTROL MODULAR

Con los microservicios, se puede escalar cada servicio de forma independiente para adaptarse a los aumentos temporales o estacionales de tráfico, realizar procesamiento por lotes y dar respuesta a otras necesidades del negocio. Un aislamiento de fallos mejorado reduce los problemas de servicio, tales como las fugas de memoria o las conexiones de base de datos abiertas, para que solo afecten al servicio específico. La escalabilidad de los microservicios complementa la flexibilidad de los servicios de nube, lo que le permite mejorar el servicio, sin interrupción, y a la vez manejar más clientes de forma simultánea.

## MÁS OPCIONES

Los métodos organizativos y las soluciones de tecnología de código abierto están liderando el mercado de los microservicios. En consecuencia, los microservicios reducen la dependencia de un proveedor y eliminan el compromiso con la tecnología a largo plazo, de modo que puede elegir las herramientas que necesita para alcanzar los objetivos de negocio y de la TI.

## CREACIÓN DE UNA BASE SÓLIDA PARA LOS MICROSERVICIOS

Para lograr el éxito con los microservicios, las organizaciones deben crear primero una base sólida para su arquitectura monolítica. La modularidad, los límites de dominio y la teoría de sistemas distribuidos fundamentales se deben considerar y establecer para aprovechar todas las ventajas de los microservicios.

Además, los microservicios ofrecen mayores ventajas con los sistemas más complejos. Aunque cada servicio es totalmente independiente, se deben cumplir ciertas necesidades operativas que incluyen capacidades tales como:

- DevOps
- PaaS
- Contenedores o VM inmutables
- Detección, registro y replicación de servicios
- Alertas y supervisión proactivas

Dado que el cumplimiento de dichos requisitos puede suponer una inversión considerable sin un retorno inmediato, quizás el uso de los microservicios no sea rentable para todos los equipos o proyectos. La evaluación de un enfoque centrado en una arquitectura monolítica garantiza que la creación de aplicaciones siga principios de diseño sólidos y que los límites de dominio estén correctamente definidos, lo que facilita una transición gradual hacia una arquitectura de microservicios si se necesita para la escalabilidad. Por ejemplo, incluso una aplicación de compras básica debe contener lo siguiente:

- Separación de responsabilidades.
- Gran cohesión y acoplamiento bajo mediante interfaces para programas de aplicación (API) bien definidas.
- Interfaces, API e implementaciones separadas, de acuerdo con la Ley de Demeter o principio de menor conocimiento.
- Diseño basado en dominios que agrupa los objetos relacionados.

Las organizaciones logran el éxito con los microservicios debido a su estructura organizativa, no a su tecnología. Los equipos flexibles con una estructura organizativa horizontal, habilidades interdisciplinares y autonomía son esenciales.

Cuando una aplicación monolítica que se debe escalar se crea de acuerdo con principios sólidos de arquitectura de software, se puede refactorizar en microservicios. El método más eficaz de refactorización consta de los siguientes pasos:

1. Identificar los límites del negocio y las diferencias semánticas del dominio de aplicación, y comenzar a descomponer cada dominio en su propio microservicio.
2. Encontrar el componente que sufre la mayoría de los cambios solicitados (como las actualizaciones de reglas de negocio asociadas a los cálculos de precios o cambios regulatorios) o al que se aplican parches a menudo para solucionar vulnerabilidades de seguridad.
3. Después de definir los microservicios básicos basados en dominios, perfeccionar las api que sirven para que los servicios interactúen. Componer estas api utilizando patrones de diseño basados en eventos, secciones, cadenas, proxies y agregadores, entre otros.

## EQUIPOS CUALIFICADOS Y ALINEADOS

Las organizaciones logran el éxito con los microservicios debido a su estructura organizativa, no a su tecnología. Los equipos flexibles con una estructura organizativa horizontal, habilidades interdisciplinares y autonomía son esenciales.

Para conseguir un equipo experto y eficaz, es necesario realinear al personal en torno a la funcionalidad en vez de a la arquitectura. Un ejemplo son los equipos de "dos pizzas" (de entre 8 y 10 personas) de Amazon, responsables de la creación y el mantenimiento del servicio. Según la Ley de Conway, las organizaciones solo pueden producir diseños que imiten la estructura de la organización. Los equipos que están segmentados (por ejemplo, en desarrollo, operaciones, control de calidad y seguridad) experimentan una limitación de la flexibilidad y retrasos en el progreso.

Establecer una práctica de DevOps antes del paso a los microservicios puede mitigar o prevenir estos problemas al determinar las estrategias de comunicación por adelantado. Esto permite evitar la creación de una SOA con fallos en lugar de una arquitectura de microservicios eficaz.

## HERRAMIENTAS DE GESTIÓN EFICACES

Además de un software bien diseñado y un equipo organizado y eficaz, para conseguir una arquitectura muy escalable se necesitan herramientas que ayuden a gestionar los servicios adicionales y los componentes de aplicaciones, lo cual incluye lo siguiente:

- Herramientas de detección y registro de servicios, como Kubernetes.
- Estándares de empaquetado para aplicaciones de organización en contenedores, como Docker, y herramientas de orquestación para replicar los contenedores a escala, como Kubernetes. OpenShift de Red Hat incluye estas dos tecnologías de código abierto probadas.
- Herramientas de creación de entornos de CI, como Jenkins o Shippable para Docker y Kubernetes.
- Herramientas de resolución de dependencia, como Nexus.
- Herramientas de resistencia y conmutación por error, incluidas las bibliotecas como Hystrix y Ribbon.
- Herramientas de supervisión del servicio, alertas y eventos, como la pila ELK (ElasticSearch, LogStash y Kibana).

## GESTIÓN DE DATOS

Otra consideración importante para la transición a los microservicios es la gestión de los datos. A diferencia de la SOA, los microservicios no comparten los datos. En su lugar, cada microservicio posee una persistencia políglota y un almacén de datos físicos independientes, lo que permite ejecutar una amplia variedad de motores de bases de datos dentro de cada microservicio. Como resultado, los almacenes de datos se pueden elegir en función del servicio, en lugar de almacenar todos los datos en un sistema de gestión de bases de datos relacionales (RDBMS) corporativo.

Sin embargo, mantener numerosas copias de una base de datos empresarial puede aumentar la complejidad y los costos de licencias. Además, puede que sea necesario alinear los almacenes de datos para mantener la coherencia. Las herramientas genéricas de extracción, transformación y carga (ETL) o de virtualización de datos pueden contribuir a la normalización de datos. Event Sourcing es un conocido patrón de diseño que ayuda a alinear los almacenes de datos para adaptarse a los cambios retroactivos.

## CONCLUSIÓN

La arquitectura de microservicios puede ofrecer muchas ventajas a las organizaciones: desde una escalabilidad independiente de diversos componentes de aplicaciones, hasta un desarrollo y un mantenimiento del software más rápidos y sencillos. No obstante, los microservicios no siempre benefician a todos los equipos o proyectos, y pueden suponer una inversión considerable sin un retorno inmediato. La transición hacia los microservicios debe ser un proceso gradual, y la refactorización de ciertas partes de las aplicaciones existentes (sin llegar a una transición total) también puede reportar beneficios. Para lograr el éxito con los microservicios, las organizaciones deben crear primero una aplicación bien diseñada de acuerdo con los estándares de la plataforma existente y, a continuación, refactorizar la aplicación en un conjunto de microservicios en función de las necesidades de su negocio. Con las personas, las herramientas y los procesos adecuados, los microservicios pueden acelerar el desarrollo y la implementación, facilitar el mantenimiento, mejorar la escalabilidad y ofrecer mayor libertad frente al compromiso con la tecnología a largo plazo.

### ACERCA DE RED HAT

Red Hat es el proveedor líder de soluciones de software de código abierto, que ha adoptado un enfoque basado en la comunidad para proporcionar tecnologías fiables y de alto rendimiento de nube, Linux, middleware, almacenamiento y virtualización. Red Hat también ofrece servicios de soporte, capacitación y consultoría que han sido premiados por su excelencia.

EUROPA, ORIENTE MEDIO Y ÁFRICA (EMEA)
00800 7334 2835
es.redhat.com
europe@redhat.com

TURQUÍA
00800-448820640

ISRAEL
1-809 449548

EAU
8000-4449549



facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

es.redhat.com
INC0336100_0216