UDEM

Universidad de Monterrey
Integración de Aplicaciones Computacionales

Tarea02 Cloud-native o Cloud-enabled

PhD. Raúl Morales Salcedo

Pablo Celedón Cabriales - 597730

Doy mi palabra de que he realizado esta actividad con integridad académica

¿Cloud-native o cloud-enabled? Análisis de una app real

En los últimos años, “migrar a la nube” dejó de ser el objetivo; lo importante es cómo se diseña y opera la aplicación una vez allá. Dos rutas típicas coexisten. Cloud-enabled es mover un sistema existente a IaaS/PaaS con cambios mínimos (rehost/replatform) para ganar elasticidad básica y servicios gestionados sin reescribir el software. Cloud-native, en cambio, parte de principios como microservicios, contenedores, infraestructura inmutable, automatización continua (CI/CD) y observabilidad profunda. La pregunta no es cuál es “mejor” en abstracto, sino cuál alinea riesgo, tiempo y valor con el contexto del producto y del equipo.

Para aclararlo, tomemos un caso real: Spotify. La plataforma evolucionó de componentes grandes a microservicios que encapsulan dominios (búsqueda, recomendaciones, catálogos, pagos) y se despliegan en contenedores orquestados. Esa decisión permite escalar solo lo que más carga recibe (por ejemplo, el servicio de recomendaciones en horas pico) y hacer lanzamientos frecuentes sin detener todo el sistema. Además, Spotify impulsó Backstage, un portal de desarrolladores que centraliza catálogos de servicios, documentación y templates de CI/CD. Aunque es una herramienta, representa una idea clave del enfoque cloud-native: si vas a tener decenas o cientos de microservicios, necesitas plataforma y estándares para que los equipos se muevan rápido sin romperse entre sí.

¿Qué gana y qué cuesta cada enfoque?

Escalabilidad y costos. Cloud-native habilita elasticidad fina: se escalan servicios específicos y se paga por uso granular. En cloud-enabled, el monolito suele escalarse como bloque, lo que puede causar sobreaprovisionamiento. En cargas variables (lanzamientos, conciertos, eventos deportivos) la capacidad de absorber picos sin sobredimensionar todo el sistema marca una diferencia tangible.

Resiliencia. El diseño nativo de nube asume que habrá fallas y prepara circuit breakers, retries, timeouts y despliegues canarios por servicio. Una caída en “pagos” no debería tumbar “reproducción de música”. En cloud-enabled, si el monolito es frágil, moverlo a máquinas más grandes en la nube puede mejorar la disponibilidad, pero no elimina acoplamientos internos.

Velocidad de entrega. Con cloud-native, equipos autónomos liberan cambios pequeños y frecuentes apoyados en pipelines, pruebas automatizadas y feature flags. El costo es la complejidad operativa: malla de servicios, seguridad de contenedores, administración de secrets, policy as code. En cloud-enabled, la curva de aprendizaje es menor; se obtienen beneficios rápidos (automatizar infraestructura, backups, monitoreo administrado) mientras se planifica una modernización por etapas.

Observabilidad y gobernanza. Microservicios requieren métricas, logs y trazas distribuidas para diagnosticar problemas de punta a punta. Sin esto, el “diseño para fallas” es teórico. También hace falta gobernanza: catálogos de APIs, estándares de despliegue y revisiones de arquitectura livianas para evitar el “espagueti distribuido”.

¿Cómo decidir?

- Elige cloud-enabled si debes migrar ya (cierre de centro de datos, fin de contrato de hardware, presión de cumplimiento) o si el producto cambia poco. Comienza con rehost/platform, adopta IaC, monitoring y backups gestionados, y define un mapa de modernización por dominios: extrae primero los módulos con mayor retorno (por ejemplo, autenticación o recomendaciones).

- Elige cloud-native si tu diferenciador depende de lanzar funcionalidades muy seguido, si esperas crecimiento acelerado o si tus SLO exigen alta disponibilidad y resiliencia distribuida. Invierte en una plataforma interna (orquestación, templates de CI/CD, golden paths), en observabilidad desde el día uno y en prácticas de seguridad para contenedores y supply chain.

Así que, cloud-native maximiza velocidad, escalabilidad y resiliencia cuando hay músculo técnico para operar la complejidad. Cloud-enabled ofrece un atajo pragmático para capturar beneficios inmediatos y reducir riesgo mientras modernizas con calma. El caso de Spotify sugiere una ruta intermedia sensata: construir una plataforma de desarrollo que haga repetible el ciclo crear-probar-desplegar, y migrar por dominios con métricas claras de valor (coste por transacción, lead time, tasa de fallos). No es una cuestión de dogmas; es de secuencia: mover primero lo urgente para capturar ahorros y seguridad, y rediseñar nativamente lo que realmente impulsa la experiencia del usuario y el negocio.

Referencias breves

— Cloud Native Computing Foundation (CNCF), “Cloud Native Definition” y reportes de adopción, 2023–2024.
— Amazon Web Services, “Migration Strategies (6/7 Rs)” y “Cloud Value Framework,” 2023–2024.
— Spotify Engineering, “Backstage: An open platform for building developer portals,” entradas técnicas 2020–2024.
— IEEE Software, artículos sobre observabilidad y análisis de causa raíz en sistemas cloud-native, 2023–2024.
— Communications of the ACM / ACM Queue, ensayos sobre confiabilidad, seguridad y microservicios a escala, 2020–2024.
— Google SRE Book (2ª ed.), capítulos sobre fiabilidad, release engineering y monitoring, 2020–2023.