Doce Factores para servicios en la nube: portabilidad, velocidad y fiabilidad

W. J. Carrillo Sandoval
Carnet: 7690-21-3740
wcarrillos1@miumg.edu.gt

Resumen
El método *twelve-factor app* define prácticas para construir servicios portables, escalables y mantenibles. Integrado con contenedores y Kubernetes, acelera despliegues, simplifica diagnósticos y mejora la paridad dev-prod. Se resumen los doce factores y su aplicación práctica.
Palabras clave: doce factores, nube, configuración, despliegue, paridad dev-prod

1. Los doce factores

1. **Código base**: una base versionada, múltiples despliegues.
2. **Dependencias**: declarar y aislar (`requirements.txt`, `package.json`).
3. **Configuración**: en variables de entorno; sin secretos en código.
4. **Servicios de apoyo**: BDs/colas/cachés como recursos adjuntos.
5. **Build, release, run**: etapas separadas e inmutables.
6. **Procesos**: sin estado; escalar horizontalmente.
7. **Asignación de puertos**: apps autocontenidas que exponen un puerto.
8. **Concurrencia**: dividir por tipos de proceso (web, worker, scheduler).
9. **Desechabilidad**: arranques rápidos y apagados limpios (señales).
10. **Paridad dev/prod**: minimizar diferencias de tiempo y herramientas.
11. **Logs**: como *streams* de eventos, centralizables y consultables.
12. **Administración**: tareas *one-off* en el mismo entorno.

2. Aplicación práctica en la nube

Contenedores para aislar dependencias; configuración vía `ENV` inyectada por el orquestador; *backing services* gestionados (DBaaS/Queue); *health checks* y *probes*; telemetría uniforme con métricas, logs y trazas.

3. Beneficios y riesgos

Beneficios: despliegues predecibles, *rollbacks* simples, menor MTTR, mayor portabilidad.
Riesgos: proliferación de variables sin gestión central, confusión entre procesos *stateless* y *stateful*; falta de SLOs y gobierno de costos.

4. Observaciones y comentarios

El mayor valor proviene de la consistencia: aplicar los doce factores en todos los servicios y entornos, complementando con SLOs, gestión de secretos y *cost governance*.

5. Conclusiones

1. Separar configuración y código mejora seguridad y portabilidad.
2. Despliegues inmutables y procesos desechables reducen el MTTR.
3. Paridad dev-prod y logs como *streams* aceleran diagnóstico y trazabilidad.

Bibliografía

1. Wiggins, A. (2011). *The Twelve-Factor App*. `https://12factor.net/`
2. Richards, M. (2015). *Microservices vs SOA*. O'Reilly.
3. Sussna, J. (2015). *Designing Delivery*. O'Reilly.