

Playbook: COBOL en 2026

Opciones para modernizar sin detener la operación

COBOL sigue sosteniendo procesos críticos de facturación, nómina, contabilidad y transacciones en bancos, aseguradoras, cooperativas y entidades públicas. El reto no es el lenguaje en sí, sino el conocimiento concentrado en pocas personas, la integración con los canales digitales y el riesgo de tocar algo que no puede fallar. Este playbook te ayuda a elegir una ruta y avanzar por etapas.

1. Por qué importa ahora

- COBOL es un estándar vivo: la edición vigente de la norma es ISO/IEC 1989:2023.
- Se sigue usando ampliamente en aplicaciones de mainframe, como el procesamiento batch y de transacciones a gran escala.
- El talento es un riesgo reconocido: en un informe de 2019 sobre sistemas legacy, la GAO (auditoría del gobierno de EE. UU.) señaló que cada vez hay menos personas con las habilidades para dar soporte a COBOL.
- En 2020, durante la pandemia, varios estados de EE. UU. reportaron escasez de programadores COBOL para sus sistemas de subsidios de desempleo.

La pregunta no es si COBOL funciona, sino cuánto riesgo aceptas al depender de pocas personas y de procesos difíciles de cambiar.

2. Señales de que es momento de actuar

- Las reglas de negocio viven en el código y en la memoria de pocas personas.
- Cada cambio regulatorio o de producto tarda demasiado por temor a romper el batch nocturno.
- Los canales digitales (app, portal, APIs) necesitan datos en línea y hoy dependen de extracciones o archivos planos.
- La documentación está desactualizada y el código es la única fuente de verdad.
- Las pruebas son manuales y los despliegues, poco frecuentes.
- Las condiciones de la plataforma (contratos, licencias, infraestructura) presionan una decisión.
- Auditoría pide trazabilidad de cambios y controles que el proceso actual cubre con dificultad.

3. Opciones y cuándo encaja cada una

Ninguna opción es mejor en abstracto. Decide por capacidad de negocio (intereses, facturación, nómina), no por el sistema completo: cada parte puede tomar una ruta distinta.

Mantener y modernizar la forma de trabajar

Cuándo encaja: la plataforma es estable y el problema principal es cómo se cambia el sistema. Se incorporan control de versiones, pruebas automatizadas, documentación de reglas y herramientas de desarrollo actuales sobre el COBOL existente.

Ten en cuenta: reduce el riesgo operativo y la dependencia de personas, pero no cambia la plataforma ni la necesidad de talento especializado.

Envolver con APIs

Cuándo encaja: los canales digitales necesitan transacciones o cálculos que ya funcionan bien. Se exponen servicios que invocan los programas existentes.

Ten en cuenta: integración rápida y reversible; la lógica sigue en COBOL y la capa de APIs debe diseñarse pensando en volumen, seguridad y ventanas batch.

Rehost (mover con cambios mínimos)

Cuándo encaja: la prioridad es salir de la infraestructura actual tocando lo menos posible el código, moviendo las cargas a un entorno compatible que ejecute COBOL.

Ten en cuenta: el código sigue siendo COBOL; hay que validar rendimiento, batch, seguridad y todo lo que el sistema de origen resolvía (transacciones, JCL, archivos).

Conversión automatizada (replatform)

Cuándo encaja: el código es consistente y quieres llegar a un lenguaje moderno, como Java o C#, sin rediseñar todo. Las herramientas convierten código y estructuras de datos.

Ten en cuenta: el resultado puede quedar como "COBOL escrito en otro lenguaje", difícil de mantener si no se refactoriza después; exige pruebas rigurosas de equivalencia.

Refactor incremental (patrón strangler fig)

Cuándo encaja: el núcleo es crítico y seguirá evolucionando. Las capacidades se reemplazan una a una detrás de una fachada que dirige cada solicitud a lo viejo o a lo nuevo, con una capa anticorrupción que aísla el modelo legacy.

Ten en cuenta: menor riesgo por paso y valor temprano, pero requiere arquitectura de transición, sincronización de datos y disciplina para llegar al retiro.

Reescribir o reemplazar con un producto

Cuándo encaja: el proceso cambió de fondo o un producto de mercado cubre la necesidad con ajustes razonables.

Ten en cuenta: mayor incertidumbre: los cálculos no documentados se pierden con facilidad y un corte único concentra el riesgo.

Combinar suele ser más sensato que elegir una sola

Por ejemplo: mejorar primero la forma de trabajar (versiones y pruebas), envolver con APIs lo que piden los canales digitales mientras se documentan las reglas, y reemplazar capacidades una a una con el patrón strangler fig. Cada paso deja la operación en un estado estable.

4. Ruta por fases

- 1

Diagnóstico

Inventario de programas, copybooks, trabajos batch y JCL, transacciones, archivos, bases de datos e interfaces. Identificación de reglas de negocio críticas; la IA puede ayudar a leer código, siempre con validación humana.

Entregable: mapa de dependencias, riesgos y ruta recomendada por capacidad de negocio.
- 2

Piloto

Una capacidad acotada y poco acoplada (por ejemplo, una consulta expuesta por API o un cálculo aislado), ejecutada en paralelo con el COBOL y comparando resultados.

Entregable: piloto en producción, criterios de equivalencia acordados y arquitectura validada.
- 3

Migración incremental

Capacidad por capacidad, en orden de valor y riesgo, con datos sincronizados, pruebas de caracterización y ventanas de corte acordadas con operación.

Entregable: en cada iteración, capacidades migradas, reglas documentadas y pruebas automatizadas.
- 4

Retiro

Cuando nada dependa del programa legacy: apagado controlado, archivo de datos históricos según los requisitos regulatorios y retiro de licencias e infraestructura.

Entregable: acta de retiro y documentación de la nueva solución.

Ejemplo ilustrativo (ficticio)

Cooperativa Demo calcula intereses con un programa COBOL nocturno. Primero expone la consulta de saldos como API para su app móvil. Luego reimplementa el cálculo en .NET, lo ejecuta en paralelo con el COBOL durante varios cierres y, cuando los resultados coinciden según los criterios acordados, retira el programa COBOL.

5. Riesgos y cómo mitigarlos

Riesgo	Cómo mitigarlo
Reglas de negocio ocultas (cálculos, redondeos, casos especiales)	Pruebas de caracterización con datos representativos y anonimizados; validación con las áreas de negocio; documentar cada regla migrada.
Precisión decimal y formatos de datos del mainframe	Pruebas específicas de precisión y de conversión de copybooks y campos empaquetados; conciliar totales entre sistemas antes de cada corte.
Procesos batch y ventanas nocturnas	Mapear el calendario y las dependencias entre trabajos; ensayar el ciclo batch completo en el entorno nuevo antes de moverlo.
Migración de datos	Cargas repetibles, sincronización durante la convivencia, conciliaciones y un plan de reversa probado.

Riesgo	Cómo mitigarlo
Operación en paralelo	Periodo acordado de ejecución en paralelo con comparación automática de resultados y criterios de salida por escrito.
Pérdida de conocimiento durante el proyecto	Sesiones de transferencia registradas y documentación viva; involucrar desde el inicio a quienes mantienen el sistema.
Cumplimiento y auditoría	Trazabilidad de cambios, evidencias de pruebas y aprobación del negocio en cada corte.

6. Checklist de preparación

- ☐ Tienes acceso al código fuente completo, incluidos copybooks y JCL.
- ☐ Hay un inventario de interfaces, archivos y procesos batch con su calendario.
- ☐ Las personas que conocen el sistema tienen tiempo asignado al proyecto.
- ☐ Hay datos de prueba representativos (anonimizados) y un entorno para comparar resultados.
- ☐ Están definidos los criterios de equivalencia: qué diferencia es aceptable y cuál no.
- ☐ Operación, riesgo y auditoría conocen el plan de convivencia y de reversa.
- ☐ Un patrocinador del negocio prioriza las capacidades a modernizar.
- ☐ Está acordado qué pasará con licencias, infraestructura y datos históricos al retirar.

7. Cómo trabajamos la modernización

En Nextsoft no ejecutamos solos la migración de sistemas legacy: la hacemos junto con aliados estratégicos especializados. El aliado realiza la migración del sistema COBOL; Nextsoft construye los módulos nuevos, las integraciones, el frontend y el QA, y es tu único punto de contacto durante todo el proyecto.

Siguiente paso

Haz el autodiagnóstico legacy gratuito: 10 preguntas sobre tecnología, operación, seguridad y negocio, con el resultado calculado en tu navegador. Si quieres conversar sobre tu caso, escríbenos.

Autodiagnóstico: nextsoftcorp.com/es/autodiagnostico-legacy

Correo: contacto@nextsoftcorp.com

Fuentes (consultadas en septiembre de 2026)

GAO-19-471, Information Technology: Agencies Need to Develop Modernization Plans for Critical Legacy Systems (2019): <https://www.gao.gov/products/gao-19-471>

Wikipedia, COBOL (norma vigente, uso en mainframe, escasez de programadores en 2020): <https://en.wikipedia.org/wiki/COBOL>
IBM, ¿Qué es COBOL?: <https://www.ibm.com/think/topics/cobol>

Microsoft Azure Architecture Center, patrón Strangler Fig: <https://learn.microsoft.com/es-es/azure/architecture/patterns/strangler-fig>

AWS Prescriptive Guidance, estrategias de migración: <https://docs.aws.amazon.com/prescriptive-guidance/latest/large-migration-guide/migration-strategies.html>