

Cobol2j

Versión 1.7

Manual de Información General

Número de publicación	GC2J-0001-01ES
Edición	Segunda edición (octubre de 2026)
Producto	Cobol2j Versión 1.7
Idioma	Español

Convención de número de formulario: GC2J-####-## identifica las publicaciones de Información General de Cobol2j. El número base se comparte en el par de idiomas; el inglés usa el sufijo numérico (GC2J-0001-01); el español usa el mismo base con sufijo ES (GC2J-0001-01ES). La revisión de cualquiera de los dos idiomas incrementa juntos los dos dígitos finales.

Nota: Antes de usar esta información y el producto al que da soporte, lea la información de los avisos y de las marcas comerciales de esta nota de edición.

Aviso de copyright

Copyright (c) 2026 Cobol2j. Todos los derechos reservados.

Esta documentación y el software que describe son propiedad de Cobol2j y están protegidos por las leyes de derecho de autor y por los tratados internacionales aplicables. No se concede ninguna licencia por el solo hecho de acceder a este documento. El uso del software por parte de un cliente se rige exclusivamente por el contrato de licencia suscrito entre ese cliente y Cobol2j. Ver [LICENSE](#) y [THIRD-PARTY-NOTICES.md](#).

EL SOFTWARE Y ESTA DOCUMENTACIÓN SE ENTREGAN «TAL CUAL», SIN GARANTÍA DE NINGÚN TIPO, EXPRESA O IMPLÍCITA, SALVO EN LA MEDIDA EN QUE EL CONTRATO DE LICENCIA APLICABLE DISPONGA OTRA COSA.

Marcas comerciales

IBM, CICS, DFSORT, IMS y VSAM son marcas comerciales o marcas registradas de International Business Machines Corporation (IBM Corp.) en numerosas jurisdicciones. En esta publicación esos nombres aparecen solo donde el lector necesita reconocerlos o escribirlos tal cual (sentencias del programa del cliente, nombres de utilidades, identificadores del producto). Cobol2j no está afiliado a IBM ni cuenta con su patrocinio o respaldo. Otros nombres de productos y servicios pueden ser marcas de sus respectivos titulares.

Contacto: hello@cobol2j.dev

Contenido

1. Acerca de este manual	1
2. Por qué migrar	3
3. Qué es Cobol2j	5
4. Para quién es	7
5. El recorrido Mainframe → Cobol2j	9
5.1 Assessment	9
5.2 Migración en el IME	10
5.3 Despliegue en el IME	10
5.4 Operación en el nuevo ambiente	10
5.5 Aplicaciones adjuntas (archivos catalogados como repositorios Java)	11
6. Los tres pilares	13
6.1 Fidelidad de la imagen de bytes EBCDIC	13
7. Java generado y comportamiento COBOL	15
8. Ambientes de ejecución y la plataforma Kubernetes	17
8.1 Batch y Online	17
8.2 Plataforma 1.7 sobre Kubernetes	17
8.3 IME y ciclo Maven	18
9. Internacionalización	19
10. Límites documentados del producto	21
11. Requisitos a alto nivel	23
12. Módulos del producto (resumen)	25
13. Publicaciones relacionadas	27

1. Acerca de este manual

Este Manual de Información General (GIM) presenta Cobol2j a evaluadores, arquitectos y gerentes técnicos. Declara qué es el producto, a quién sirve, el **recorrido Mainframe → Cobol2j** en cinco pasos, qué cubre a alto nivel y dónde empiezan los manuales detallados.

No es una guía de instalación ni un manual de operación. Esos documentos están en [§13 Publicaciones relacionadas](#).

Los hechos de esta edición reflejan Cobol2j **1.7** según la documentación del repositorio del producto (octubre de 2026).

2. Por qué migrar

Las aplicaciones de mainframe que aún sostienen la lógica de negocio central enfrentan un problema estructural que reescribir desde cero rara vez resuelve de forma limpia:

- **Competencias.** La experiencia en Enterprise COBOL y en operación de mainframe es escasa y se retira. Contratar y retener a quienes puedan cambiar con seguridad programas batch, de transacciones online y de archivos indexados certificados a escala es cada año más difícil.
- **Costo.** El hardware, las licencias de software y la operación especializada siguen subiendo, mientras las mismas cargas podrían ejecutarse en plataformas Java de uso general que la organización ya opera.
- **Dependencia de hardware.** La aplicación permanece atada a la planificación de capacidad, a las herramientas y a los centros de datos del mainframe, aunque el negocio ya no necesite ese pie de fábrica para esas cargas.
- **Riesgo.** La lógica crítica vive en programas que pocas personas pueden modificar con seguridad. El mantenimiento diferido, las interfaces congeladas y los diseños de datos frágiles aumentan la exposición operativa y de cumplimiento.

Modernizar esas aplicaciones significa llevarlas a un entorno de ejecución contemporáneo **sin descartar** la lógica certificada, los diseños de registro EBCDIC ni los comandos de transacciones online y el lenguaje de control JCL que los equipos ya conocen. Cobol2j está construido para ese camino: rehospedar sobre Java 21, Spring Boot y Kubernetes preservando la semántica de datos y de programa del mainframe.

3. Qué es Cobol2j

Cobol2j es una **plataforma de rehosting** de aplicaciones COBOL online y batch de mainframe sobre **Java 21 + Spring Boot + Kubernetes**.

Permite **ejecutar en Java** programas y datos que hoy viven en un mainframe: **archivos EBCDIC indexados y secuenciales**, pantallas **BMS/3270**, programas **COBOL** online y batch, y datos **SQL**. No es un emulador: traduce los artefactos del mainframe a artefactos Java y web nativos, preservando la semántica de los datos (EBCDIC, COMP-3, longitud fija, colación de claves, decimal exacto).

Dos productos bajo la misma plataforma:

Producto	Rol
Cobol2j Online	Regiones que atienden transacciones de terminal (semántica pseudo-conversacional de transacciones online).
Cobol2j Batch	Colas de trabajos que ejecutan el JCL original (clases, prioridades, spool, consola), con Spring Batch por debajo.

El COBOL sigue con EXEC CICS y el JCL sigue siendo JCL: esa es la sintaxis que la organización ya tiene, y Cobol2j la ejecuta tal cual.

La versión **1.7** entrega los tres pilares (datos, pantallas, traductor), un **IME** web multiusuario (Integrated Migration Environment), los dos ambientes de ejecución con **EXEC SQL** integrado transaccionalmente con la E/S de archivos, e instalación con **Helm sobre Kubernetes**.

4. Para quién es

Cobol2j sirve a organizaciones que necesitan:

- Salir del hardware de mainframe para cargas seleccionadas y dejar de usar COBOL como lengua del desarrollo **nuevo**.
- Conservar la lógica certificada de batch, de transacciones online, de archivos indexados y de SQL estático **sin reescribirla** a Spring idiomático.
- Ser dueñas del Java certificado después de migrar.
- Consultar los datos migrados con herramientas ordinarias de base de datos y de BI, sin una utilidad de mainframe.
- Operar el runtime sobre **su propio** cluster de Kubernetes.
- Adjuntar aplicaciones Java **nuevas** que lean y escriban los mismos archivos catalogados mediante una API Spring Data y el contrato de datos publicado.

5. El recorrido Mainframe → Cobol2j

El camino del producto desde un portafolio de mainframe hasta un sistema Cobol2j en ejecución son cinco pasos. Cada paso tiene un procedimiento documentado; esta sección declara la secuencia y el límite de cada uno. El detalle está en los manuales citados al final de cada subsección.

Paso	Nombre	Dónde ocurre	Resultado
1	Assessment	En las instalaciones (<code>java -jar</code> o <code>mvn cobol2j:assess</code>)	Migrabilidad medida del portafolio; inventario, semáforo, grafos, informe y láminas
2	Migración en el IME	Perspectiva Desarrollo del IME (o <code>mvn cobol2j:migrate / certify</code>)	Java y HTML generados; informe por archivo; propiedad certificada del Java
3	Despliegue en el IME	IME Desarrollo → Desplegar (o <code>mvn cobol2j:deploy</code>)	Clases, pantallas, JCL y tarjetas de control en la load library del sistema; TRANSID y catálogo de archivos en la región destino; resultado auditado
4	Operación en el nuevo ambiente	Perspectivas Sistemas, Operación y Seguridad del IME; terminal Online; cola / spool / consola batch; visor de datos	Operación diaria de Online, batch y datos sin el mainframe
5	Aplicaciones adjuntas	Aplicaciones Spring Boot nuevas con <code>cobol2j-spring-data-vsam</code> y el contrato de datos publicado	Desarrollo Java nuevo contra los mismos archivos catalogados como repositorios

Los cinco pasos, en orden:

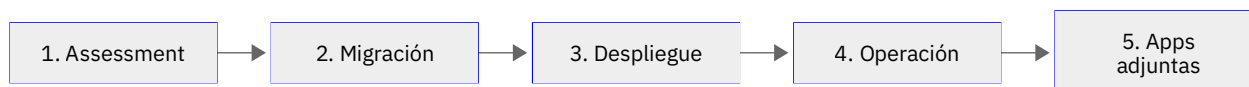


Figura 1.

Los goals de Maven usan el **mismo motor de migración** que el IME (`cobol2j-migration`). Tras `certify`, el Java pertenece al equipo y una corrida posterior de `migrate` no lo regenera salvo opciones de forzado. El mantenimiento de fase 2 de programas certificados puede hacerse en cualquier IDE Java; el IME sigue siendo el lugar de la **operación** (cola, spool, consola, terminal, datos).

5.1 Assessment

Antes de comprometerse a una migración, Cobol2j mide el portafolio **corriendo el traductor en seco** sobre el código de la organización — no con una estimación heurística.

Propiedad	Hecho
Entrega	Un <code>java -jar</code> firmado que se ejecuta en las instalaciones , o <code>mvn cobol2j:assess</code> .
Red	No abre ninguna conexión de red. El fuente no sale del sitio por el assessment mismo.
Salida	Inventario, semáforo por objeto (automática / asistida / bloquea / fuera de alcance), grafos, censo de verbos, informe y láminas ejecutivas/técnicas desde un solo modelo.
Prosa	Texto determinista del producto a partir de campos medidos — no un modelo de lenguaje externo.

Los criterios del semáforo coinciden con la aceptación del proyecto: automática exige **0 TODO y 0 WARNING**.

Los artefactos que definen el **ambiente** (CSD, catálogo de datasets, DDL de SQL, red del planificador, definiciones del gestor de seguridad, SIT, clases de trabajo y similares) **no** forman parte del portafolio de código y hay que pedirlos aparte; sin ellos el sistema migrado no arranca. Ver GC2J-0003-01ES Guía de assessment.md.

5.2 Migración en el IME

La migración se hace en la perspectiva **Desarrollo** del IME: seleccionar el proyecto y ejecutar **Migrar**. El mismo motor que usa `mvn cobol2j:migrate` recorre copybooks, mapas BMS y programas COBOL y produce el Java y el HTML equivalentes, más un informe por archivo: **GENERADO, OMITIDO, SOBRESCRITO, CERTIFICADO** o **ERROR**. Las construcciones no soportadas se clasifican **TODO** (bloquea), **WARNING** (traducida, puede diferir del mainframe) o **INFO**. El informe completo queda en `migracion.md` / `migracion.json` y se ve en la consola de migración del IME. El procedimiento paso a paso, desde preparar las fuentes hasta la puesta en producción, está en GC2J-0010-01ES Guía de migración de aplicaciones.

El IME también ofrece:

- Un comparador fuente↔generado (COBOL a la izquierda, Java a la derecha) para revisar un programa.
- Clones Git por usuario con commit, publicar y sincronizar.
- Estructura de proyecto para COBOL, copybooks, BMS, JCL, PROC y tarjetas de control (JCL/PROC/CTL **no** se traducen; se despliegan tal cual).

La certificación (`mvn cobol2j:certify` o el cierre equivalente de la fase 1) registra la propiedad del Java generado. Ver GC2J-0007-01ES Manual de usuario.md §§5–7 y §13.

5.3 Despliegue en el IME

Desplegar instala en la load library del **sistema destino** (volumen compartido: directorio local o exportación NFS en Kubernetes): archivos `.class`, pantallas, JCL, procedimientos y tarjetas de control, de forma atómica. Cada región e iniciador que sirve el sistema detecta el cambio con un escaneo periódico del volumen y lo toma sin reiniciar.

Las dos mitades de un despliegue no van al mismo lado:

- Los miembros de la load library son **del sistema** (compartidos por todas las regiones e iniciadores de ese sistema).
- El TRANSID y el catálogo de archivos van solo a la **región elegida**; hay que repetirlos para que una región nueva atienda las mismas transacciones.

Si el programa tiene `EXEC SQL`, el despliegue pide al motor del sistema destino que prepare cada sentencia (analizar y resolver nombres, sin ejecutar) — el equivalente Cobol2j de un `BIND` de mainframe. Los avisos no impiden instalar. Todo despliegue queda auditado (quién, cuándo, qué, a qué sistema y región, con qué resultado). Ver GC2J-0007-01ES Manual de usuario.md §9 y GC2J-0008-01ES Guía de administración del sistema.md §16.3.

5.4 Operación en el nuevo ambiente

El trabajo diario tras el despliegue usa las **cuatro perspectivas** del IME (el rol determina qué acciones aparecen):

Perspectiva	Roles típicos	Para qué
Desarrollo	DESARROLLO, ADMINISTRACION	Árbol de proyecto, editor, migrar, comparar, Git, desplegar.
Sistemas	Los cuatro roles; escritura principalmente ADMINISTRACION	Sistemas, regiones y sus catálogos, catálogo de datos, iniciadores.
Operación	OPERACION, ADMINISTRACION; SEGURIDAD en lectura	Arrancar/parar regiones e iniciadores; cola, spool, consola, terminal; lo denegado.
Seguridad	Escritura SEGURIDAD; los demás en lectura	Cuentas del IME y su auditoría; usuarios, grupos y perfiles de Cobol2j Seguridad.

Online. Desde Operación (o la URL de la región): el terminal HTML con los mismos campos protegidos y atributos de color del mapa BMS original. Las teclas (ENTER, PF1–PF24, PA1–PA3, CLEAR) impulsan la semántica pseudo-conversacional de transacciones online; el estado de sesión vive en PostgreSQL (C2JTERMID), así que las regiones con varias instancias no exigen sesión pegajosa.

Batch. Desde Operación: someter un trabajo (pegar/subir o lector interno INTRDR); **cola** por CLASS= / PRTY= atendida por iniciadores libres; **spool** de DD SYSOUT=; **consola** con identificadores estables (C2Jnnnn); mensajes al operador (estilo WTOR) dejan el trabajo en pausa hasta que se responde desde Operación.

Datos. El **visor de datos** del IME decodifica un archivo del catálogo con su copybook de vista (modos tabla o hex), carga datasets EBCDIC, exporta CSV y puede crear vistas ASCII para BI (vascii_*) en el esquema del sistema. Los datasets catalogados viven en el **catálogo del sistema**, no en el proyecto de migración. Ver GC2J-0007-01ES Manual de usuario.md §§4, 8, 11, 12 y GC2J-0008-01ES Guía de administración del sistema.md §10.

5.5 Aplicaciones adjuntas (archivos catalogados como repositorios Java)

Cuando el portafolio ya corre en Cobol2j, las aplicaciones Java/Spring Boot **nuevas** pueden usar los mismos archivos catalogados sin aprender COBOL. Ese camino es distinto del Java que produce el traductor.

Capacidad	Hecho
Dependencia	Módulo publicado cobol2j-spring-data-vsam (más cobol2j-data-platform y cobol2j-core).
Forma de la API	VsamRepository es a un archivo catalogado lo que JpaRepository es a una tabla (findById, save, tipos de paginación, DataAccessException).
Camino de acceso	La aplicación no abre la base de datos. Pide los datos a una región Online que ya tiene el archivo abierto (URL, SYSID y token configurados).
Aprovisionamiento	Operación da de alta la aplicación como conexión de la región (SYSID + token) en el IME — el mismo modelo que una región hermana.
Origen del layout	El diseño del registro (copybook, longitud, code page, índices alternos) sale del descriptor del archivo en la región al arrancar — la clase de la aplicación no declara más que el nombre del archivo y la región.
Consultas	Solo por clave o por índice alterno declarado. El análisis SQL completo usa el contrato de datos publicado : vistas relacionales tipadas en el esquema bi_<sistema> (ver GC2J-0006-01ES §4).

Capacidad	Hecho
Unidad de trabajo	Una aplicación adjunta no entra en la unidad de trabajo distribuida de la región ni del paso batch. Cada operación suya se confirma sola. Dentro de una región, el SQL y los archivos con RECOVERY (BACKOUTONLY) o ALL se confirman y se deshacen juntos; en un paso batch, como en z/OS, el SQL se deshace al abendar y los archivos VSAM no.

Regla práctica documentada en la Guía de acceso a datos: **el repositorio para llegar a un registro y escribirlo; el contrato de datos publicado para preguntar**. Ver GC2J-0005-01ES Guía de acceso a datos para aplicaciones adjuntas.md y GC2J-0006-01ES Guía de arquitectura técnica.md §4.

6. Los tres pilares

#	Pilar	Qué hace
1	Datos (archivos EBCDIC)	Almacena los registros de archivo como imagen de bytes EBCDIC cruda en PostgreSQL (bytea). Solo decodifica al ver o exportar con un copybook.
2	Pantallas (BMS)	Analiza mapas BMS y genera pantallas HTML5 para las regiones Online.
3	Traductor (COBOL → Java)	Traducción ahead-of-time: una clase Java imperativa por programa. Lo no soportado emite un marcador visible <code>// TODO cobol2j</code> : y falla de forma ruidosa — nunca Java incorrecto en silencio.

6.1 Fidelidad de la imagen de bytes EBCDIC

Migrar datos de mainframe suele fallar en el primer paso de conversión. Un paso EBCDIC→ASCII o UTF-8 corrompe:

- **Decimal empaquetado (COMP-3)** — dos dígitos por byte, signo en el nibble bajo.
- **Decimal zoned** — signo incrustado en el nibble alto del último byte.
- **Binario (COMP / COMP-4 / COMP-5)** y coma flotante hexadecimal (HFP, COMP-1 / COMP-2).
- **Orden de claves KSDS** — la colación EBCDIC difiere de ASCII; los dígitos ordenan distinto.

Cobol2j no convierte esos campos al ingerir. Cada registro de archivo se guarda como la **imagen de bytes exacta** en PostgreSQL bytea. La colación de claves KSDS sigue el orden de bytea sin signo, que coincide con el orden EBCDIC. La decodificación se difiere hasta que se aplica un copybook para visualización, exportación o una vista de BI (`vascii_*`). Los mismos bytes pueden releerse con otro copybook (`REDEFINES`) sin volver a migrar el archivo.

Capacidades de soporte a nivel de producto:

- Catálogo de datos global estilo IDCAMS: organización, RECFM, largo de registro, clave e índices alternos independientes del copybook.
- Vistas ASCII para BI que decodifican el bytea al leer.
- La API **adjunta** de repositorios Spring Data descrita en §5.5.
- Commit y rollback juntos del SQL y de los archivos recuperables dentro de una región. En un paso batch, el SQL se deshace al abendar y los archivos VSAM no, como en z/OS.

7. Java generado y comportamiento COBOL

El traductor emite Java imperativo ordinario que sigue comportándose como COBOL allí donde la lógica de negocio lo exige. Las características siguientes están documentadas en la Guía de arquitectura técnica (GC2J-0006-01ES) y el README del producto; no son afirmaciones aspiracionales.

Característica	Comportamiento en el Java generado / runtime
Decimal exacto	La aritmética usa <code>CobolDecimal</code> (escala del campo receptor, truncado o <code>ROUNDED</code>). El código generado no usa <code>double</code> para numéricos COBOL.
COMP-3 y USAGE relacionados	Decimal empaquetado, zoned, binario y coma flotante hexadecimal (COMP-1/COMP-2) los atienden los códecs EBCDIC; en el límite Java los valores se exponen como <code>BigDecimal</code> .
Imagen física del registro	Las escrituras de campo mantienen sincronizada la imagen de bytes; el <code>MOVE</code> de grupo opera sobre bytes crudos con la semántica de relleno del mainframe.
Índices de tabla 1-based	Los subíndices de <code>OCCURS</code> siguen la indexación COBOL (basada en 1). El acceso fuera de rango sigue el estilo <code>NOSSRANGE</code> del mainframe dentro del área de datos, con aviso en el log.
Semántica de transacciones online	Los programas traducidos conservan <code>EXEC CICS</code> (BMS, pseudo-conversación con <code>COMMAREA</code> , <code>EIB</code> , <code>LINK/XCTL</code> , <code>E/S</code> de archivos, colas <code>TS/TD</code> , <code>ENQ/DEQ</code> , <code>SYNCPPOINT</code> y comandos relacionados).
Semántica JCL	El batch ejecuta el JCL original : <code>DD</code> , <code>SYSIN</code> , <code>PARM</code> , <code>IF/COND</code> , <code>PROC/INCLUDE/symbolics</code> , <code>GDG</code> y utilidades habituales, con cola de trabajos estilo mainframe, spool y consola.
EXEC SQL	SQL estático y dinámico con fidelidad de variables host; <code>COMMIT/ROLLBACK</code> atómico con la <code>E/S</code> de los archivos recuperables dentro de una región; en un paso batch el SQL se deshace al abendar y los archivos VSAM no, como en <code>z/OS</code> .

Lo no soportado se marca y falla de forma visible: el traductor emite un marcador `TODO`, `WARNING` o `INFO` y el informe de migración por archivo los enumera todos. Qué está soportado, construcción por construcción, se detalla en la Referencia de Cobol2j, GC2J-0009-01ES Referencia de Cobol2j.md; su §14 enumera los límites.

8. Ambientes de ejecución y la plataforma Kubernetes

8.1 Batch y Online

Ambiente	Resumen
Batch	Ejecuta el JCL original: DD, SYSIN, PARM, IF / COND, PROC/INCLUDE/symbolics, GDG, utilidades (IEFBR14, IEBGENER, DFSORT, IDCAMS y otras). Ofrece cola de trabajos estilo mainframe, spool y consola de operador.
Online	Región sobre Tomcat: terminal web, transacciones por TRANSID, catálogos de programas y archivos en caliente, estado pseudo-conversacional en PostgreSQL (apto para varias instancias; no se exige sesión pegajosa para ese estado).

8.2 Plataforma 1.7 sobre Kubernetes

Una instalación de estilo productivo usa Helm:

- Chart `cobol2j-plataforma` (una vez por instalación) y chart `cobol2j-sistema` (una vez por sistema).
- Namespace por sistema; TLS de punta a punta en el Ingress; volumen compartido `ReadWriteMany` como fuente de verdad del PDS.
- El agente del sistema actúa como controlador de Kubernetes (Server-Side Apply para Deployments de región e iniciador).
- Overlays de valores para EKS, AKS y GKE.
- Las regiones nacen elásticas con HPA (típicamente 1–3 réplicas).

Cómo se relacionan el chart de plataforma, los charts de sistema y el volumen compartido:

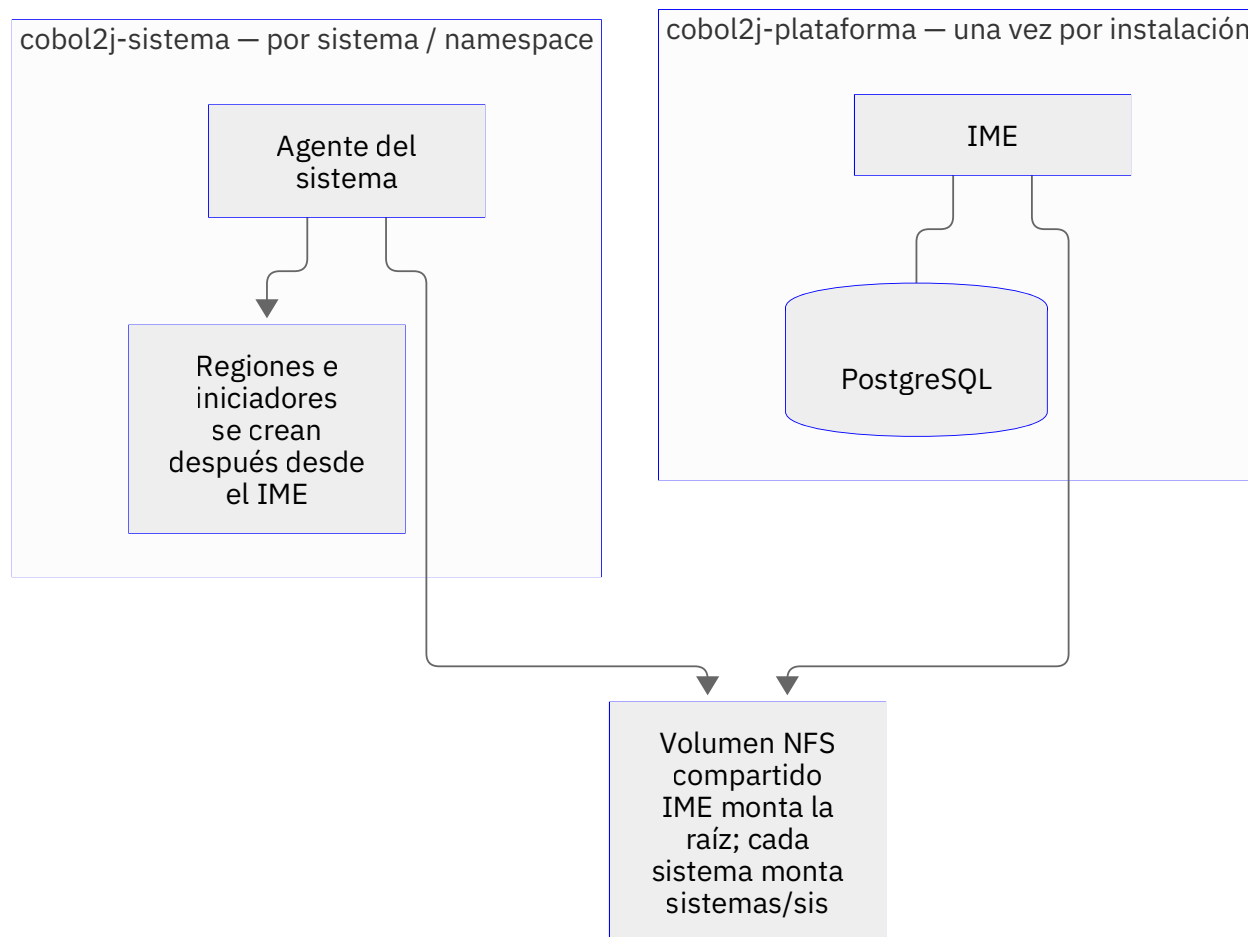


Figura 2.

La instalación y la operación día a día están en la Guía de instalación (GC2J-0002-01ES) y en la Guía de administración del sistema (GC2J-0008-01ES), no aquí. El recorrido de §5 supone una plataforma ya instalada.

8.3 IME y ciclo Maven

El **IME** es el entorno web multiusuario de migración y operación cuyas cuatro perspectivas se resumen en §5.4.

El plugin Maven cubre `cobol2j:assess`, `cobol2j:migrate`, `cobol2j:certify` y `cobol2j:deploy` para línea de comandos / CI sin la UI del IME. Tras `certify`, el Java es del equipo y `migrate` no lo regenera.

9. Internacionalización

El idioma de superficie del producto es **inglés por omisión, español bajo pedido** (`--lang es / -Dcobol2j.lang=es / assessment -Dcobol2j.assess.lang=es`). Una sola dimensión de idioma (`en | es`) cubre los entregables del assessment, los marcadores del traductor que el informe reenvía y la UI del IME donde está localizada.

Los nombres de nivel `WARNING` e `INFO` no se renombran (las corridas antiguas que usaban `AVISO` siguen abriéndose).

Este GIM se publica en ambos idiomas como números de formulario GC2J-0001-01 (EN) y GC2J-0001-01ES (ES).

10. Límites documentados del producto

Lo siguiente son **decisiones documentadas del producto**:

Límite	Significado
IMS	Base de datos jerárquica y gestor de transacciones: fuera de alcance (0 %) salvo que un compromiso concreto lo requiera.
FTP	Utilidad no implementada; los decks que la necesitan no corren (falla explícita / fuera de alcance).
Assembler del cliente	Los subprogramas en Assembler quedan fuera de alcance por decisión (por ejemplo COBDATFT y MVSWAIT en la aplicación de ejemplo CardDemo).
Adjunto ≠ misma UOW	Una aplicación adjunta nueva Spring Data habla con una región para la E/S de archivo; no entra en la unidad de trabajo de la región ni del paso batch. El SQL y los archivos recuperables se confirman juntos dentro de una región (GC2J-0009-01 §7.4).
Construcciones de cola larga	Lo no soportado emite TODO/WARNING/INFO, enumerados en el informe de migración por archivo.

Las regiones Online con varias instancias también documentan límites sobre el CWA local al proceso, el empuje de pantalla de START ... TERMID y las tareas conversacionales que no migran de instancia — ver la Guía de arquitectura técnica (GC2J-0006-01ES).

11. Requisitos a alto nivel

Entorno mínimo para construir y ejecutar Cobol2j (detalle en la Guía de instalación, GC2J-0002-01ES):

Componente	Nivel
JDK	21 (JDK completo; el despliegue compila con <code>javac</code>)
Maven	3.9+
PostgreSQL	16+
Docker	Para pruebas de integración (Testcontainers) y construcción de imágenes
Kubernetes	Cluster con Helm para una instalación completa; <code>kind</code> / Docker Desktop alcanzan para pruebas locales

Resumen de stack: Java 21, Spring Boot 4 / Spring Framework 7, PostgreSQL, Kubernetes (Helm, cliente fabric8), cert-manager, volumen RWX compartido, Thymeleaf + HTMX (IME).

Reactor groupId `dev.cobol2j`, versión **1.7.0** para esta edición.

12. Módulos del producto (resumen)

Dieciséis módulos Maven. Mapa a alto nivel:

Área	Módulos (ejemplos)
Datos y códecs	cobol2j-core, cobol2j-data-platform, cobol2j-dataview, cobol2j-spring-data-vsam
Pantallas	cobol2j-screen-platform
Traducir y migrar	cobol2j-translator, cobol2j-migration, cobol2j-maven-plugin
Runtime	cobol2j-runtime, cobol2j-batch-runtime, cobol2j-online-runtime
Plataforma	cobol2j-kubernetes
Assessment	cobol2j-assessment, cobol2j-assessment-cli
IME / ejemplos	cobol2j-ime, cobol2j-examples

Responsabilidades completas de cada módulo: GC2J-0006-01ES Guía de arquitectura técnica.md.

13. Publicaciones relacionadas

Formulario / ruta	Título	Audiencia
GC2J-0001-01 / GC2J-0001-01 General Information.md	General Information Manual (inglés)	Misma audiencia, inglés
GC2J-0002-01ES / GC2J-0002-01ES Guía de instalación.md	Guía de instalación	Quienes instalan la plataforma
GC2J-0002-01 / GC2J-0002-01 Installation Guide.md	Installation Guide (inglés)	Misma audiencia, inglés
GC2J-0003-01ES / GC2J-0003-01ES Guía de assessment.md	Guía de assessment	Quien ejecuta el assessment
GC2J-0003-01 / GC2J-0003-01 Assessment Guide.md	Assessment Guide (inglés)	Misma audiencia, inglés
GC2J-0004-01ES / GC2J-0004-01ES Guía del administrador de seguridad.md	Guía del administrador de seguridad	Administradores de seguridad
GC2J-0004-01 / GC2J-0004-01 Security Administrator's Guide.md	Security Administrator's Guide (inglés)	Misma audiencia, inglés
GC2J-0005-01ES / GC2J-0005-01ES Guía de acceso a datos para aplicaciones adjuntas.md	Guía de acceso a datos para aplicaciones adjuntas	Autores de aplicaciones Java/Spring Boot adjuntas
GC2J-0005-01 / GC2J-0005-01 Data Access Guide for Attached Applications.md	Data Access Guide for Attached Applications (inglés)	Misma audiencia, inglés
GC2J-0006-01ES / GC2J-0006-01ES Guía de arquitectura técnica.md	Guía de arquitectura técnica	Quienes mantienen o extienden el producto
GC2J-0006-01 / GC2J-0006-01 Technical Architecture Guide.md	Technical Architecture Guide (inglés)	Misma audiencia, inglés
GC2J-0007-01ES / GC2J-0007-01ES Manual de usuario.md	Manual de usuario	Programadores y operadores (IME)
GC2J-0007-01 / GC2J-0007-01 User's Manual.md	User's Manual (inglés)	Misma audiencia, inglés
GC2J-0008-01ES / GC2J-0008-01ES Guía de administración del sistema.md	Guía de administración del sistema	Administradores y programadores de sistemas
GC2J-0008-01 / GC2J-0008-01 System Administration Guide.md	System Administration Guide (inglés)	Misma audiencia, inglés
GC2J-0009-01ES / GC2J-0009-01ES Referencia de Cobol2j.md	Referencia de Cobol2j	Migradores, programadores del Java generado y operadores de JCL
GC2J-0009-01 / GC2J-0009-01 Cobol2j Reference.md	Cobol2j Reference (inglés)	Misma audiencia, inglés
GC2J-0010-01ES / GC2J-0010-01ES Guía de migración de aplicaciones.md	Guía de migración de aplicaciones	Programadores de aplicaciones e ingenieros de migración
GC2J-0010-01 / GC2J-0010-01 Application Migration Guide.md	Application Migration Guide (inglés)	Misma audiencia, inglés

Formulario / ruta	Título	Audiencia
GC2J-0011-01ES / GC2J-0011-01ES Mensajes y códigos.md	Mensajes y códigos	Operadores, programadores y administradores que buscan un mensaje o un código
GC2J-0011-01 / GC2J-0011-01 Messages and Codes.md	Messages and Codes (inglés)	Misma audiencia, inglés
deploy/README.md	Charts de despliegue	Preparación del cluster

IBM, CICS, DFSORT, IMS y VSAM son marcas comerciales o marcas registradas de International Business Machines Corp.

Fin de GC2J-0001-01ES