

Presentación para perfiles técnicos, de dirección y de negocio. Recorre la arquitectura completa, la seguridad, los flujos de negocio de punta a punta con sus llamadas HTTP, las pruebas y los pendientes de decisión. Todas las referencias externas están al final.

AGENDA

Qué vamos a revisar

Parte	Tema	Qué resuelve
1	Negocio y principios	Qué áreas existen y por qué se diseñó así
2	Arquitectura	Piezas, datos, seguridad y mensajería
3	Operación	Errores, monitoreo, trazas y despliegue
4	Flujos de negocio	Ventas, facturación, compras, documentos y reportes
5	Código y cierre	Llamadas reales, pruebas, pendientes y referencias

Ruta de la presentación. La parte 4 es la más práctica: cada flujo muestra los pasos en orden, desde obtener el token hasta cerrar el proceso.

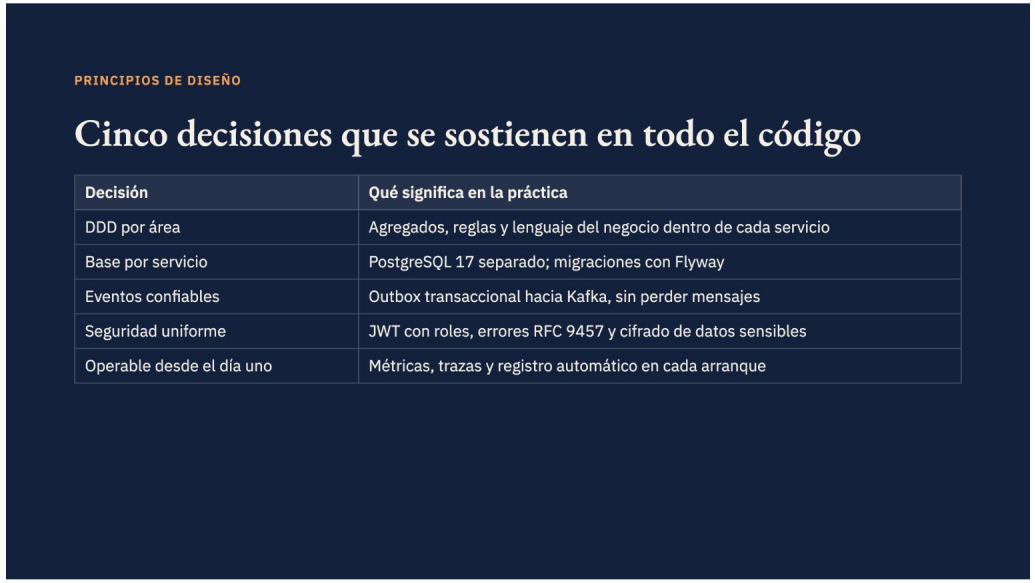
NEGOCIO

Un servicio por área, cada uno dueño de sus datos

Área	Servicio	Responsabilidad
Ventas y clientes	clientes-ms, pedidos-ventas-ms	Prospectos, clientes, cotizaciones y pedidos
Finanzas	facturacion-pagos-ms, contabilidad-ms	Comprobantes, pagos y pólizas contables
Operaciones y compras	inventario-operaciones-ms, compras-proveedores-ms	Existencias, lotes, órdenes y proveedores
Legal	documentos-cumplimiento-ms	Contratos y permisos con versiones y retención
Dirección	reportes-ejecutivos-ms	Indicadores consolidados y reportes cerrados
Acceso	usuarios-accesos-ms	Usuarios, roles y emisión del token

Cada servicio tiene su base de datos. Ningún servicio lee la base de otro.

Nueve servicios de área más el gateway y el registro. Cada área puede cambiar su servicio sin tocar los demás, siempre que respete el contrato de su API y sus eventos.



Estas cinco decisiones explican casi todo lo demás de la presentación. Si una se rompe, la plataforma deja de ser predecible.

ARQUITECTURA GENERAL

Entrada única, servicios aislados, eventos confiables

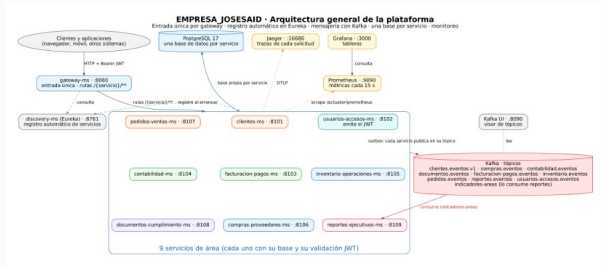


Diagrama completo en el repositorio: docs/diagramas/arquitectura-general.png. Cada servicio tiene además su propio diagrama en su carpeta docs/diagramas. Las flechas punteadas son registro, consultas y métricas; las sólidas son tráfico y datos.

MAPA DE PUERTOS

Dentro de Docker cada servicio usa su propio puerto

Grupo	Componentes	Puerto en tu Mac
Entrada	gateway-ms	8080
Registro	discovery-ms (Eureka)	8761
Servicios de área	clientes 8101 · usuarios 8102 · facturación 8103 · contabilidad 8104 · inventario 8105 · compras 8106 · pedidos 8107 · documentos 8108 · reportes 8109	8101 a 8109
Monitoreo	Prometheus · Grafana	9090 · 3000
Trazas	Jaeger	16686
Mensajería	Kafka UI	8090

Dentro de la red empresa-net, cada servicio responde en su puerto interno de 8080 (o 8081 en los de área). Los puertos de la tabla son sólo para el host.

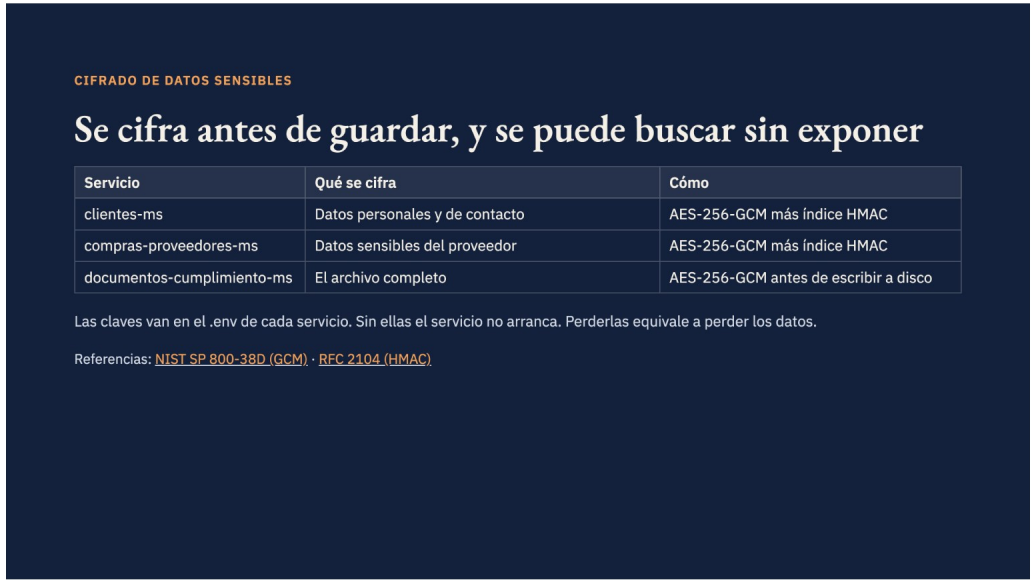
Los puertos del host son los que se usan para pruebas directas. Para producción se expone sólo el gateway. Hoy Kafka UI, Prometheus, Grafana y Jaeger están abiertos en el host porque el entorno es local.



Detalle clave para operar: después de reiniciar el registro, los servicios tardan hasta un minuto en reaparecer. Durante ese tiempo el gateway puede responder 404. No es un error del servicio.



Riesgo a conocer: el secreto es compartido por todos los servicios. Quien lo tenga puede firmar tokens válidos para cualquiera. Rotarlo implica reiniciar todos los servicios. La alternativa de validación centralizada en el gateway sigue pendiente de decisión.



El índice HMAC permite buscar por RFC o correo sin guardar el valor original. Es una decisión de privacidad: la búsqueda exacta funciona, la búsqueda parcial no.



Por qué importa: el esquema siempre coincide con el código y se puede reconstruir desde cero. Las bases no se comparten, así que un cambio en un servicio no rompe a otro.



Pregunta típica de dirección: ¿qué pasa si se cae Kafka? La respuesta es que los cambios se guardan, los eventos se acumulan y se envían al recuperarse. Lo que sí puede ocurrir es un evento repetido, por eso los consumidores deben ser idempotentes.

TÓPICOS DE KAFKA

Quién publica, quién consume

Tópico	Quién publica	Quién consume
clientes.eventos.v1	clientes-ms	Ninguno todavía
compras.eventos	compras-proveedores-ms	Ninguno todavía
contabilidad.eventos	contabilidad-ms	Ninguno todavía
documentos.eventos	documentos-cumplimiento-ms	Ninguno todavía
facturacion-pagos.eventos	facturacion-pagos-ms	Ninguno todavía
inventario.eventos	inventario-operaciones-ms	Ninguno todavía
pedidos.eventos	pedidos-ventas-ms	Ninguno todavía
reportes.eventos	reportes-ejecutivos-ms	Ninguno todavía
usuarios-accesos.eventos	usuarios-accesos-ms	Ninguno todavía
indicadores-areas	Las áreas, por API o Kafka	reportes-ejecutivos-ms

Cada evento usa como clave el id de su agregado, así los eventos de un mismo agregado llegan en orden. Un tópico aparece en Kafka UI cuando recibe su primer evento.

ERRORES

Cada error es explicable y tiene su código

HTTP	Excepción de dominio	Cuándo ocurre
400	SolicitudInvalida	Datos mal formados o fuera de formato
404	RecursoNoEncontrado	El id no existe
409	Conflicto · OptimisticLocking	Duplicado o cambio simultáneo
410	CotizacionVencida	La cotización ya no se puede aceptar
422	ReglaDeNegocio	La operación viola una regla del dominio

Todas las respuestas de error siguen RFC 9457: type, title, status, detail e instance.

Las reglas viven en el dominio, no en el controlador. Por eso un error 422 significa que el negocio dijo no, y un 400 que la solicitud estaba mal armada.

OBSERVABILIDAD

Métricas, tableros y trazas de cada solicitud

Pieza	Qué responde	Dónde se ve
Prometheus	Métricas de cada servicio cada 15 s	localhost:9090/targets
Grafana	Tablero JVM (Micrometer, ID 4701)	localhost:3000
Jaeger	Camino de cada solicitud y tiempo por servicio	localhost:16686

Verificado: una traza del gateway a usuarios-accesos-ms con 6 spans y 2 servicios.

Referencias: Micrometer · Prometheus · Grafana · OTLP · Jaeger

Las trazas se muestran al 100 % de las solicitudes en este entorno. En producción conviene bajar el muestreo para controlar costo. Jaeger guarda las trazas en memoria: al reiniciarlo se pierden.



Compose no admite depends_on entre archivos separados. Por eso el orden se documenta en el README raíz y en el de cada servicio. Cada servicio también puede levantarse solo desde su carpeta.



El token dura lo que indica expiresIn. Todavía no hay renovación ni cierre de sesión: son decisiones pendientes que conviene revisar antes de producción.

FLUJO 1 · VENTAS, DE PROSPECTO A ENTREGA

Ocho llamadas del prospecto a la entrega

Paso	Llamada	Resultado
1	POST /clientes-ms/v1/prospectos	Prospecto NUEVO
2	PATCH /clientes-ms/v1/prospectos/{id}/estatus	CONTACTADO, luego CALIFICADO
3	POST /clientes-ms/v1/prospectos/{id}/conversion	Cliente con contacto principal
4	POST /pedidos-ventas-ms/v1/cotizaciones	Cotización en borrador
5	POST .../cotizaciones/{id}/envio y .../aceptacion	Aceptada, o 410 si venció
6	POST .../cotizaciones/{id}/pedido	Pedido creado desde la cotización
7	POST .../pedidos/{id}/confirmacion	Pedido CONFIRMADO
8	POST .../pedidos/{id}/entrega	Pedido ENTREGADO

Rutas completas a través del gateway: http://localhost:8080 más la ruta de la tabla.

Cada paso cambia el estado de un agregado y deja su evento en el outbox. Si el cliente pide cancelar, POST .../pedidos/{id}/cancelacion con un motivo: CANCELACION_CLIENTE, ERROR_CAPTURA o DEVOLUCION_TOTAL. Una devolución posterior a la entrega usa POST .../devolucion.



Impacto para dirección: el cierre de periodo es el punto de no retorno contable. Antes de cerrar, la balanza debe revisarse. Las pólizas reversadas quedan registradas, no se borran.

FLUJO 3 · COMPRAS, INVENTARIO Y TRAZABILIDAD

De la orden de compra al lote que se puede rastrear

Paso	Llamada	Resultado
1	POST /compras-proveedores-ms/v1/proveedores	Proveedor dado de alta
2	POST /compras-proveedores-ms/v1/ordenes-compra	Orden con líneas por SKU
3	POST .../ordenes-compra/{id}/aprobacion	Orden aprobada
4	POST .../ordenes-compra/{id}/recepciones	Cantidades recibidas
5	POST /inventario-operaciones-ms/v1/movimientos	ENTRADA con su lote
6	GET /inventario-operaciones-ms/v1/lotes/{lote}/trazabilidad	Historial completo del lote

Producción: POST /inventario-operaciones-ms/v1/ordenes-operacion y, al terminar, POST .../ordenes-operacion/{id}/cierre con la cantidad real.

El lote obligatorio se define en cada artículo. Antes de usar el flujo en operación, conviene confirmar en pruebas qué pasa si una entrada llega sin lote.

FLUJO 4 · DOCUMENTOS DE CUMPLIMIENTO

Subir, versionar, vigilar el vencimiento y retirar

Paso	Llamada	Resultado
1	POST /documentos-cumplimiento-ms/v1/documentos (multipart: datos y archivo)	Documento cifrado en disco
2	POST .../documentos/{id}/versiones (multipart: archivo)	Nueva versión, sin modificar la anterior
3	GET .../documentos/por-vencer?dias=30	Lista de lo que vence pronto
4	GET .../documentos/{id}/versiones/{n}/descarga	Archivo de la versión n
5	POST .../documentos/{id}/eliminacion con motivo	Eliminación registrada

Retención por tipo: contrato 10 años, permiso 5 años, registro legal 10 años. Pendiente de confirmación legal.

Cada día a las 7:00 corre una tarea que avisa de los documentos por vencer. La purga física al vencer la retención sigue pendiente de confirmación legal: hoy sólo se registra la eliminación.



Para dirección: el cierre convierte el reporte en un documento de referencia. Si un indicador cambia después, el cambio se refleja en el siguiente periodo, no en el cerrado. Acceso por área pendiente: hoy sólo el rol DIRECCION ve todo.

PASO A PASO · TOKEN Y PRIMERA LLAMADA

```
# 1. Obtener el token
curl -s -X POST http://localhost:8080/usuarios-accesos-ms/v1/auth/login \
  -H "Content-Type: application/json" \
  -d '{"correo":"admin@empresa.mx","password":"<su-contraseña>"}'
# respuesta: accessToken, tokenType, expiresIn

# 2. Guardar el valor de accessToken
TOKEN="<accessToken>"

# 3. Llamar una ruta protegida
curl -s "http://localhost:8080/clientes-ms/v1/clientes?rfc=ABC850101XY1" \
  -H "Authorization: Bearer $TOKEN"
```

Sin token o vencido: 401. Con token pero sin el rol requerido: 403.

Este bloque es lo mínimo para empezar: un login y una consulta. Los flujos completos usan exactamente el mismo encabezado Authorization en cada llamada.

PASO A PASO · DE PROSPECTO A COTIZACIÓN

```
# 1. Crear el prospecto
curl -s -X POST http://localhost:8080/clientes-ms/v1/prospectos \
  -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{"nombre":"Distribuidora del Norte","correo":"compras@norte.mx","rfc":"ABC850101XY1","fuente":"REFERIDO"}'
# 2. Convertir en cliente (id del paso 1)
curl -s -X POST http://localhost:8080/clientes-ms/v1/prospectos/{id}/conversion \
  -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{"razonSocial":"Distribuidora del Norte SA de CV","rfc":"ABC850101XY1",
    "contactoPrincipal":{"nombre":"Ana López","correo":"ana.lopez@norte.mx","rol":"COMPRAS"}}'
# 3. Crear la cotización (clienteId del paso 2)
curl -s -X POST http://localhost:8080/pedidos-ventas-ms/v1/cotizaciones \
  -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
  -d '{"clienteId":"{clienteId}","canal":"WEB","moneda":"MXN","tasaIva":"IVA_GENERAL",
    "vigenteHasta":"2026-11-30","lineas":[{"productoId":"{productoId}","cantidad":2,"precioUnitario":1200.00}]}'
```

Los ids se toman de la respuesta de cada paso. Después vienen envío, aceptación y conversión a pedido, como se muestra en el flujo de ventas. Reemplazar los valores entre llaves y el token antes de ejecutar.



Ejemplo ilustrativo del formato RFC 9457. El valor exacto de type depende de cada error del servicio. Lo importante para integrar sistemas es el status y el type, no el texto de detail.



Las pruebas de integración encontraron fallas reales que se corrigieron: en documentos, Hibernate intentaba actualizar una FK bloqueada por un trigger; en reportes, una prueba seguía esperando un 400 para PDF cuando ese formato ya se soporta. Por eso se corren contra bases y brokers reales.



Las decisiones de esta lista no bloquean el uso local. Sí bloquean un despliegue productivo, en especial la retención documental, la validación de tokens y la protección de paneles.



Un despliegue local de más de 50 contenedores se levantó y se probó de punta a punta. Eso demuestra la integración; no reemplaza las pruebas de integración ni la revisión de seguridad para producción.

REFERENCIAS · 1 DE 2

Documentación oficial de las piezas principales

Tema	Enlace
Spring Boot	spring.io/projects/spring-boot
Spring Cloud Gateway	docs.spring.io/spring-cloud-gateway/reference/
Netflix Eureka	github.com/Netflix/eureka/wiki
Spring Security, JWT	docs.spring.io/spring-security/.../jwt.html
RFC 7519 (JWT)	rfc-editor.org/rfc/rfc7519
RFC 9457 (errores)	rfc-editor.org/rfc/rfc9457
NIST SP 800-38D (GCM)	csrc.nist.gov · SP 800-38D
Transactional outbox	microservices.io/patterns/data/...
Apache Kafka	kafka.apache.org/documentation/

Las referencias de la segunda parte cubren base de datos, migraciones, monitoreo, trazas, contenedores y pruebas.

REFERENCIAS · 2 DE 2

Datos, monitoreo, contenedores y pruebas

Tema	Enlace
Flyway	documentation.red-gate.com/flyway
PostgreSQL 17	postgresql.org/docs/17/
Micrometer	micrometer.io/docs
Prometheus	prometheus.io/docs/
Grafana	grafana.com/docs/grafana/latest/
OpenTelemetry (OTLP)	opentelemetry.io/docs/specs/otlp/
Jaeger	jaegertracing.io/docs/
Kafka UI	github.com/provectus/kafka-ui
Docker Compose	docs.docker.com/compose/
Testcontainers y JUnit 5	testcontainers.com · junit.org (JUnit 5)

Las referencias son documentación oficial de cada tecnología. Las versiones que usa el proyecto están en cada pom.xml y en las imágenes de los docker-compose.



Cierre de la presentación. Todo lo que se mostró tiene respaldo en el repositorio: código, README por servicio, manual HTML y diagramas en docs/diagramas.