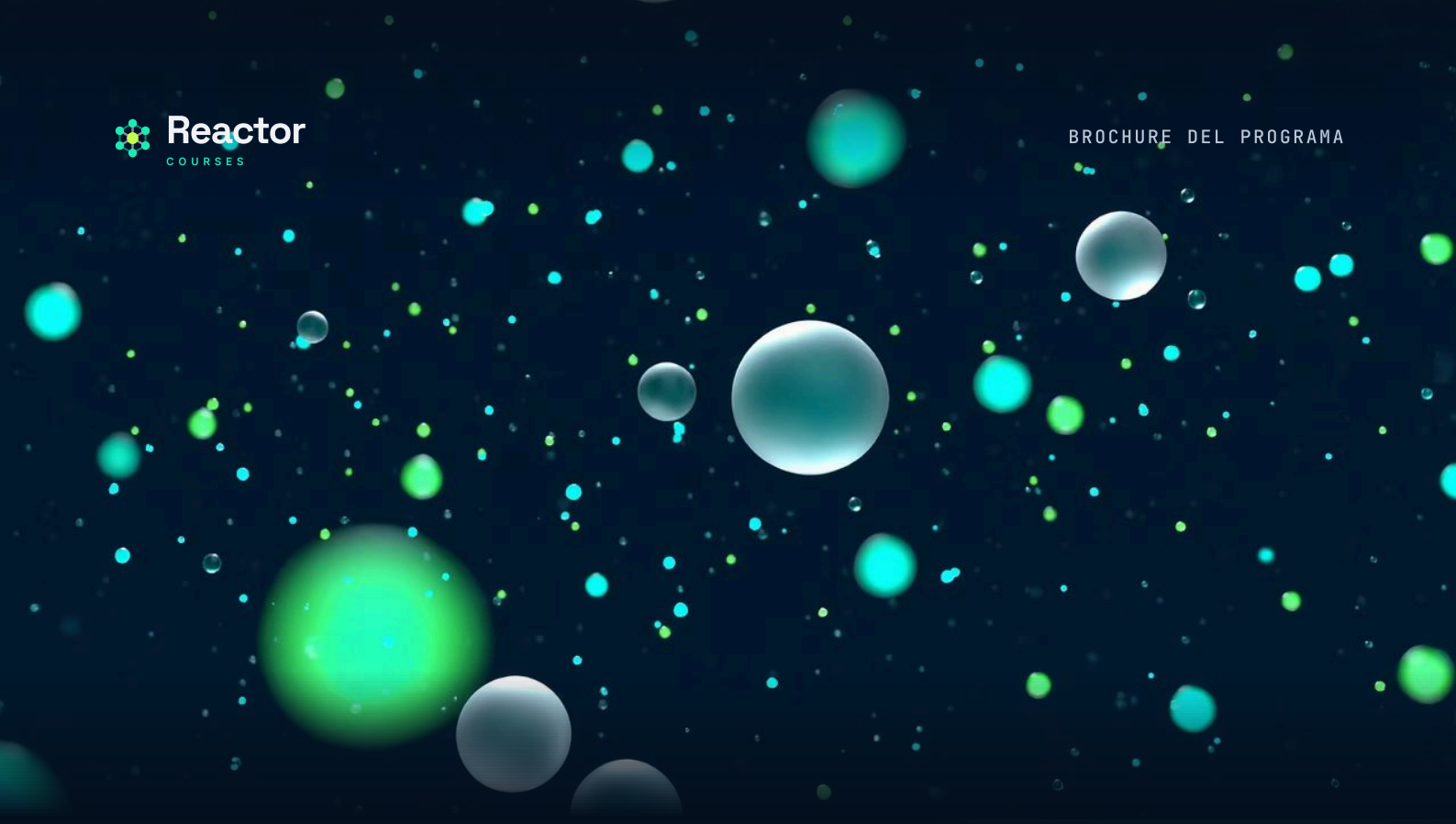


Reactor COURSES

BROCHURE DEL PROGRAMA

ARQUITECTURA E INGENIERÍA · INTEGRACIÓN Y DATOS

Arquitectura orientada a eventos

Diseñar con hechos, no con llamadas

Modela un dominio de negocio con eventos y opera el flujo que lo sostiene: quién es dueño de cada evento, cómo evoluciona su esquema, cómo se reconstruye el estado y qué pasa de verdad cuando un proceso muere a mitad del camino.

Lo que vas a lograr Aprendes a conectar sistemas por medio de eventos —«se creó un pedido», «se pagó una factura»— en lugar de llamadas directas, y a que el flujo resista las fallas.

- Event storming y DDD
- Kafka y RabbitMQ
- Outbox e idempotencia
- Event sourcing y CQRS
- Esquemas y compatibilidad
- Relectura y reproceso

horas en vivo	sesiones	semanas	puntos donde vas a matar procesos
60	20	10	5

INICIO
Por confirmar

PRECIO
US$360 pago único · USD

MODALIDAD
Clases en vivo por Zoom · Avanzado

reactorcourses.com/programas/event-driven-architecture

EL PROGRAMA

Publicar eventos es fácil. Diseñar un dominio con ellos y operarlo sin perder pedidos no lo es.

Modela un dominio de negocio con eventos y opera el flujo que lo sostiene: quién es dueño de cada evento, cómo evoluciona su esquema, cómo se reconstruye el estado y qué pasa de verdad cuando un proceso muere a mitad del camino. Un programa de **60 horas** para quienes ya usan una cola o Kafka y sufren duplicados, desorden o esquemas rotos, y para quienes evalúan event sourcing y quieren saber también cuándo no conviene.

Duración	60 horas · 20 sesiones	Calendario	10 semanas · 2 sesiones por semana de 3 h
Modalidad	Clases en vivo por Zoom	Grabaciones	Disponibles por 12 meses
Credenciales	3 certificados con insignia de Reactor Courses	Nivel	Avanzado
Inicio	Por confirmar	Horario	Martes y jueves, 19:00–22:00 (GMT-5)
Precio	US$360 · pago único		

Muchos equipos llegan a los eventos por la infraestructura —se instaló Kafka— y descubren después que el problema era de diseño: eventos sin dueño, esquemas que rompen a quien los lee, procesos que mueren a mitad del camino y proyecciones que nadie sabe reconstruir. Este programa empieza por el dominio y termina operando el flujo.

01

Modelar y asegurar la entrega

Dominio, contextos, broker, garantías y esquemas

Event storming con el negocio, contextos delimitados con un dueño por evento, la elección entre log y cola según el dominio, y las garantías de entrega probadas matando procesos, con la frontera exacta de lo que el broker promete. Los esquemas evolucionan con un registro que rechaza lo incompatible.

- Event storming, contextos delimitados y agregados
- Kafka y RabbitMQ elegidos por flujo, no por moda
- Bandeja de salida, consumidores idempotentes y captura de cambios
- Registro de esquemas con reglas de compatibilidad

02

Reconstruir y operar

Event sourcing, CQRS, procesos largos y operación del flujo

Event sourcing donde aporta y no donde no, proyecciones que se reconstruyen desde cero después de corregir un error, eventos guardados que cambian de forma sin reescribir la historia, procesos largos con compensación y la operación diaria: mensajes venenosos, reproceso, trazas a través del broker y pruebas que no dependen de la suerte.

- Almacén de eventos sobre PostgreSQL y proyecciones CQRS
- Versionado de eventos guardados y reconstrucción de proyecciones
- Coreografía u orquestación, y consistencia percibida por el usuario
- Cola de fallidos, trazas asíncronas y pruebas estables

> Exactamente una vez existe, pero termina en el borde del broker. Desde ahí en adelante, la garantía la escribes tú.

PARA QUIÉN ES

¿Es el programa indicado para ti?

Está pensado para desarrolladores que ya trabajaron con alguna cola o broker y quieren diseñar sistemas en los que los mensajes sostienen el negocio.

Es para ti si

- ✓ Desarrolladores y arquitectos que ya usan una cola o Kafka y sufren duplicados, desorden o esquemas rotos.
- ✓ Equipos que migran integraciones síncronas a eventos.
- ✓ Arquitectos que evalúan event sourcing y CQRS para un dominio concreto.
- ✓ Líderes técnicos que tienen que decidir dueños de eventos entre equipos.
- ✓ Ingenieros que operan flujos asíncronos y no saben cómo reprocesar sin miedo.

Te conviene otro programa

- – Quien no haya trabajado con ninguna cola o broker: los fundamentos de mensajería se dan por sabidos.
- – Quien busque la teoría formal de consistencia y consenso: aquí se usan las garantías, no se demuestran.
- – Quien quiera un curso de administración de Kafka: el broker se opera lo necesario para diseñar bien.
- – Quien busque procesamiento analítico de flujos: ventanas y agregaciones para análisis no son el tema.
- – Quien ya cursó un programa de sistemas distribuidos o de plataforma de datos: cerca de un tercio de este programa —log y cola, idempotencia y bandeja de salida, esquemas, sagas, mensajes fallidos y eventos publicados— toca problemas que esos programas tratan. Lee en las preguntas frecuentes la que trata ese caso.
- – Quien no tenga un equipo con 16 GB de memoria o no pueda dedicar entre 10 y 12 horas por semana durante 10 semanas.

EL RESULTADO

Terminas con un sistema de pedidos y despacho que sobrevive a procesos muertos y se reconstruye desde su historia.

Cada laboratorio deja una pieza del sistema del caso con la prueba de que hace lo que dice. Juntas forman el proyecto que defiendes en la última sesión.

01 **Modelo del dominio** Event storming del proceso de pedido a entrega, contextos delimitados y agregados con sus invariantes justificadas.

02 **Eventos bien diseñados** Eventos elegidos entre notificación, transferencia de estado y dominio, con un consumidor nuevo que no necesita llamar de vuelta.

03 **Broker elegido por flujo** El mismo flujo en Kafka y en RabbitMQ, con la pérdida de orden demostrada y la elección justificada.

04 **Ningún despacho duplicado** Bandeja de salida y consumidores idempotentes probados matando procesos en cinco puntos del flujo.

05 **Esquemas que no rompen** Registro de esquemas que rechaza el cambio incompatible mientras el consumidor antiguo sigue leyendo.

06 **Event sourcing con concurrencia** Agregado de pedido sobre PostgreSQL que no produce estados imposibles con comandos simultáneos.

07 **Proyecciones que se reanudan** Tres proyecciones del mismo flujo que retoman desde su posición sin perder ni repetir eventos.

08 **Historia que cambia de forma** Evento versionado con transformación al leer y mil pedidos históricos reconstruidos idénticos.

09 **Reconstrucción sin efectos repetidos** Proyección de facturación corregida y reconstruida desde cero sin reenviar ninguna factura.

10 **Proceso largo y usuario tranquilo** Despacho por coreografía y por orquestación con cancelación a mitad, y un cliente que nunca ve desaparecer su pedido.

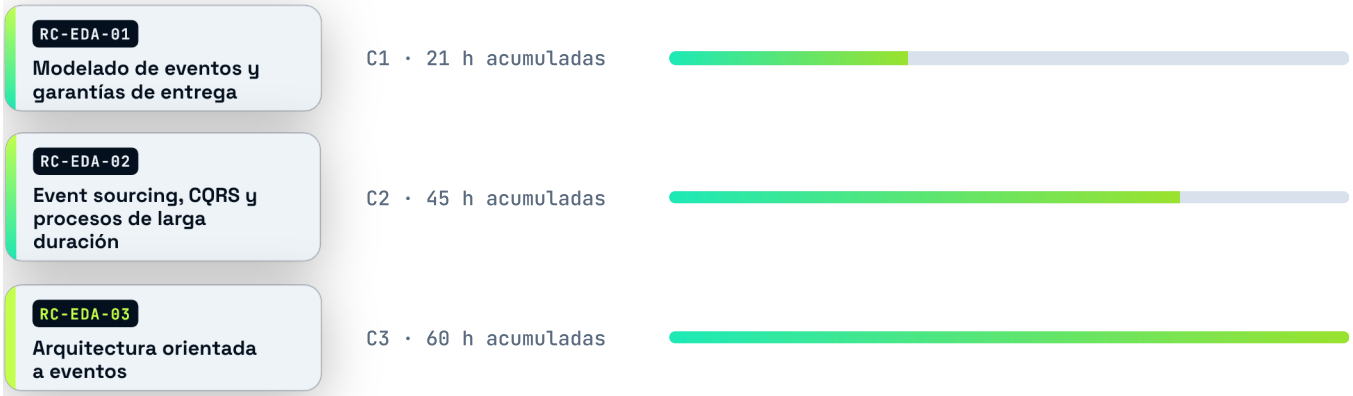
11 **Operación del flujo** Mensaje venenoso aislado y reprocesado, y una traza de punta a punta a través del broker con alertas por desfase.

12 **Pruebas estables** Batería de pruebas con broker real sin pausas fijas, estable diez veces seguidas en la tubería.

RUTA DE CERTIFICACIÓN

3 credenciales progresivas, no una al final.

Cada etapa cierra un bloque completo: primero modelar y asegurar la entrega, después reconstruir el estado y coordinar procesos largos, y al final operar el flujo. Son credenciales de Reactor Courses, no certificaciones de terceros.



CERTIFICACIÓN INTEGRAL

Event-Driven Architecture — Diseñar con hechos, no con llamadas (60 horas)

Requiere entregar el sistema de pedidos y despacho con sus pruebas, la reconstrucción completa de una proyección y la defensa con procesos muertos en vivo. Los certificados son emitidos por Reactor Courses y acreditan las horas y competencias del programa.

Cada certificado de Reactor Courses viene con su insignia, aparte de cualquier certificación oficial.

Cualquiera puede comprobar un certificado de Reactor Courses con su código en reactorcourses.com/verificar.

TEMARIO

Las 20 sesiones, abiertas y sin letra chica.

60 horas repartidas en 6 módulos. Así sabes qué verás, qué practicarás y qué entregarás en cada una de las 10 semanas.

MÓDULO 1 **MODELAR EL DOMINIO CON EVENTOS** — 12 h · 4 sesiones

01 **Hechos, no llamadas**

evento, comando y mensaje: diferencias que cambian el diseño · un hecho es inmutable y está en pasado · acoplamiento temporal de las llamadas síncronas · el caso: pedidos y despacho de un operador de última milla · qué gana y qué pierde un sistema que reacciona a hechos · el entorno local: broker, base de datos y servicios en contenedores

LABORATORIO: Levantar el sistema de pedidos del caso, convertir tres llamadas síncronas entre servicios en eventos y medir qué pasa con un pedido cuando el servicio de despacho está caído antes y después del cambio

02 **Event storming**

event storming de panorama y de diseño · eventos de dominio, comandos, actores y políticas · puntos calientes y preguntas abiertas · sistemas externos y modelos de lectura · conducir la sesión con expertos del negocio · del tablero al código sin perder el lenguaje

LABORATORIO: Conducir un event storming del proceso de pedido a entrega con un experto del negocio simulado y entregar el tablero digital con eventos, comandos, políticas y cinco puntos calientes resueltos

03 **Límites de contexto y dueño del evento**

contexto delimitado como frontera del lenguaje · mapa de contextos y relaciones entre equipos · un evento tiene un solo dueño · agregados como frontera de consistencia · invariantes que obligan a un agregado más grande · traducción de eventos entre contextos

LABORATORIO: Dividir el dominio del caso en contextos delimitados con un dueño por evento, diseñar los agregados de pedido y de despacho con sus invariantes, y justificar por qué cada invariante queda dentro de un solo agregado

04 **Tipos de evento y su contrato interno**

evento de notificación · evento con transferencia de estado · evento de dominio · cuánta información poner en un evento · eventos internos frente a eventos publicados a otros contextos · nombres de eventos en el lenguaje del negocio

LABORATORIO: Diseñar los eventos del caso eligiendo para cada uno entre notificación, transferencia de estado o dominio, y demostrar con un consumidor nuevo que el diseño elegido no le obliga a llamar de vuelta al productor

MÓDULO 2 EL BROKER AL SERVICIO DEL DOMINIO

9 h · 3 sesiones

05 **Log o cola, elegidos por el dominio**

registro particionado con Apache Kafka y cola con RabbitMQ · retención y relectura frente a consumo y descarte · clave de orden elegida desde el agregado · cuántas particiones y por qué · consumidores competidores y su efecto en el orden · un broker de cada tipo en el mismo sistema

LABORATORIO: Implementar el mismo flujo de despacho sobre Kafka y sobre RabbitMQ, demostrar en cuál se pierde el orden por pedido con consumidores competidores, y elegir el broker de cada flujo del caso con su justificación

06 **Garantías aplicadas: idempotencia, outbox y dónde termina el exactamente una vez**

al menos una vez como punto de partida realista · productor transaccional y lectura confirmada en Kafka · el límite del exactamente una vez cuando la salida es otro sistema · bandeja de salida transaccional desde el agregado · consumidor idempotente con tabla de procesados · captura de cambios con Debezium como alternativa a la bandeja · matar procesos a mitad del flujo como prueba

LABORATORIO: Publicar los eventos del pedido con bandeja de salida —y en una variante con captura de cambios de Debezium— y consumirlos de forma idempotente, y demostrar matando procesos en cinco puntos distintos del flujo que ningún pedido se despacha dos veces ni queda sin despachar

07 **Esquemas, registro y compatibilidad**

Avro, Protobuf y JSON Schema: qué cambia en la práctica · registro de esquemas con Apicurio · compatibilidad hacia atrás, hacia adelante y completa · campo nuevo, campo eliminado y cambio de tipo · consumidores con versiones distintas en paralelo · por qué no el registro de Confluent en este programa

LABORATORIO: Registrar los esquemas del caso en Apicurio con reglas de compatibilidad, hacer tres cambios y demostrar que el registro rechaza el incompatible mientras un consumidor antiguo sigue leyendo los compatibles

MÓDULO 3 EVENT SOURCING Y CQRS

15 h · 5 sesiones

08 Agregados y almacén de eventos

el estado como resultado de aplicar eventos · almacén de eventos sobre PostgreSQL · concurrencia optimista por versión del agregado · comandos que se validan contra el estado reconstruido · instantáneas y cuándo hacen falta · por qué no un almacén de eventos con licencia propietaria en este programa

LABORATORIO: Implementar el agregado de pedido con event sourcing sobre PostgreSQL y demostrar que dos comandos simultáneos sobre el mismo pedido no producen un estado imposible

09 Proyecciones CQRS

separar el modelo de escritura del de lectura · proyección construida a partir de eventos · varias proyecciones del mismo flujo para usos distintos · posición de la proyección y su reanudación · proyecciones en otra base de datos · costo de mantener cada proyección

LABORATORIO: Construir tres proyecciones del caso —seguimiento del cliente, tablero de despacho y facturación— desde el mismo flujo de eventos, y demostrar que una proyección detenida retoma desde su posición sin perder ni repetir eventos

10 Versionar eventos ya guardados

los eventos guardados no se editan · transformación al leer: upcasting · evento nuevo frente a versión nueva del mismo evento · migración de un flujo completo y su riesgo · pruebas con eventos históricos reales · deuda de versiones viejas en el almacén

LABORATORIO: Cambiar la forma de un evento de pedido con dos años de historia simulada mediante transformación al leer, y demostrar que el estado reconstruido de mil pedidos históricos es idéntico antes y después

11 Reproceso y reconstrucción de proyecciones

corregir un error de proyección reconstruyéndola desde cero · reconstrucción en paralelo y cambio de tráfico · tiempo de reconstrucción y su límite práctico · efectos externos que no deben repetirse en un reproceso · reproceso parcial por agregado · verificar la proyección reconstruida

LABORATORIO: Introducir un error en la proyección de facturación, corregirlo y reconstruir la proyección desde el almacén de eventos en paralelo a la vieja, demostrando que no se reenvía ninguna factura y midiendo el tiempo de reconstrucción

12 Cuándo no usar event sourcing

dominios donde la historia no tiene valor · costo operativo y curva de aprendizaje del equipo · consultas que el modelo de eventos vuelve difíciles · borrar datos personales en un registro inmutable · borrado criptográfico y sus límites · event sourcing en un solo agregado, no en todo el sistema

LABORATORIO: Evaluar cinco subdominios del caso y decidir en cuáles usar event sourcing con un registro de decisión, e implementar el borrado de los datos personales de un cliente en el almacén de eventos verificando que ya no se pueden reconstruir

MÓDULO 4 PROCESOS LARGOS Y CONSISTENCIA PERCIBIDA

9 h · 3 sesiones

13 Coreografía u orquestación de un proceso del dominio

el proceso de despacho como flujo de varios contextos · coreografía: cada contexto reacciona a hechos · orquestación con Temporal: un dueño del proceso · compensaciones expresadas en el lenguaje del negocio · visibilidad del proceso para soporte · criterio para elegir según el dominio

LABORATORIO: Implementar el proceso de despacho una vez por coreografía y otra con Temporal, cancelar un pedido a mitad del proceso en ambas, y elegir para el caso la variante que deja el estado más fácil de explicar a soporte

14 **Consistencia eventual frente al usuario**

el usuario no sabe que el sistema es asíncrono · leer lo que uno acaba de escribir · esperar la proyección, devolver el comando o mostrar lo pendiente · versiones en la respuesta para consultas posteriores · notificaciones en tiempo real al cliente · medir el desfase que el usuario percibe

LABORATORIO: Implementar en la app de seguimiento la lectura de lo propio escrito con la versión del agregado, y demostrar con un desfase de proyección inyectado de cinco segundos que el cliente nunca ve su pedido desaparecer

15 **Eventos hacia afuera: CloudEvents y AsyncAPI**

evento interno frente a evento de integración publicado · traducir eventos de dominio a eventos de integración · CloudEvents 1.0.2: atributos y formatos · documentar el flujo con AsyncAPI 3 · qué no debe salir del contexto · un consumidor externo que no controlas

LABORATORIO: Publicar los eventos de entrega a un socio externo como CloudEvents traducidos desde los eventos de dominio, documentados en AsyncAPI 3, y demostrar que un cambio interno del agregado no altera el evento publicado

MÓDULO 5 OPERAR EL FLUJO — 9 h · 3 sesiones

16 **Mensajes fallidos, reproceso y relectura operativa**

error transitorio frente a mensaje venenoso · reintentos con retroceso en el consumidor · cola de mensajes fallidos y su revisión · reprocesar después de corregir · relectura desde una posición del registro · manual de operación para un flujo detenido

LABORATORIO: Introducir un mensaje venenoso y una falla transitoria en el flujo de despacho, demostrar que el venenoso va a la cola de fallidos sin detener el resto, y reprocesarlo después de corregir el consumidor siguiendo un manual escrito

17 **Observabilidad asíncrona**

trazas que atraviesan el broker con OpenTelemetry · convenciones semánticas de mensajería · desfase de consumidores como señal · correlación del pedido de punta a punta · edad del evento más viejo sin procesar · tablero de un flujo, no de un servicio

LABORATORIO: Instrumentar el flujo con OpenTelemetry a través de Kafka y RabbitMQ, seguir un pedido de punta a punta en una sola traza y configurar alertas por desfase y por edad del evento más viejo, verificadas con un consumidor detenido

18 **Probar sistemas asíncronos**

pruebas que esperan con sondeo en lugar de pausas fijas · Testcontainers para pruebas con broker real · pruebas de agregado dado, cuando, entonces · pruebas de proyección con flujos grabados · contratos de eventos entre equipos · pruebas de caos del flujo en la tubería

LABORATORIO: Escribir la batería de pruebas del caso —agregado, proyección y flujo completo con broker real en Testcontainers— sin ninguna pausa fija, y demostrar que corre estable diez veces seguidas en la tubería

MÓDULO 6 PROYECTO INTEGRADOR — 6 h · 2 sesiones

19 **Evolución en producción simulada y relectura completa**

cambio de esquema con consumidores en paralelo · error de negocio descubierto tarde · corrección y reconstrucción completa de una proyección · verificación del estado contra el almacén de eventos · comunicación del reproceso a los equipos afectados · registro de lo hecho para la defensa

LABORATORIO: Ejecutar sobre el sistema completo un cambio de esquema en producción simulada y la corrección de un error de negocio con reconstrucción completa de una proyección, sin repetir efectos externos y con el estado verificado contra el almacén

20 **Defensa del sistema**

modelo del dominio y dueño de cada evento · garantías declaradas y dónde terminan · event sourcing donde se usó y donde no · proceso largo y consistencia percibida · operación del flujo y pruebas · preguntas cruzadas del grupo

`LABORATORIO:` Defender ante el grupo el sistema de pedidos y despacho matando procesos en vivo en los puntos que elija el grupo, y sustentar las decisiones de modelo, broker y garantías con los datos del propio laboratorio

---

`LO QUE VAS A USAR`

Software libre elegido por la decisión que resuelve, sin licencias restrictivas.

Brokers

Apache Kafka · RabbitMQ

Datos y eventos

PostgreSQL como almacén de eventos · Debezium · Apicurio Registry

Procesos

Temporal

Contratos

CloudEvents 1.0.2 · AsyncAPI 3

Operación y pruebas

OpenTelemetry · Testcontainers

Modelado

Event storming en tablero digital

Los logos son marcas de sus respectivos dueños. Se muestran solo para identificar las herramientas que se usan en el programa y no implican aval, patrocinio ni afiliación.

CÓMO FUNCIONA

Clases en vivo, un dominio real y cada garantía probada matando procesos.

01 **Clases en vivo por Zoom** 20 sesiones de 3 horas, 2 veces por semana. Se modela en vivo con el grupo y se revisa el código de cada laboratorio.

02 **Todo queda grabado** Si te pierdes una sesión o vives en otro huso horario, la grabación queda disponible. Acceso por 12 meses.

03 **30% concepto / 70% laboratorio** El modelado se hace con el grupo y enseguida se implementa sobre el sistema del caso.

04 **Las garantías se prueban, no se asumen** Cada laboratorio de garantías mata procesos en puntos elegidos del flujo y verifica el estado final.

05 **Dónde termina cada promesa** Se cita lo que el broker promete y se marca el punto exacto donde la garantía pasa a ser responsabilidad de tu código.

06 **También cuándo no** Una sesión entera para decidir dónde event sourcing no conviene, con el borrado de datos personales incluido.

Horarios · Primer grupo

Martes	19:00 - 22:00
Jueves	19:00 - 22:00

Horario referido a GMT-5 (Perú, Colombia, Ecuador y Panamá; en Ciudad de México es una hora menos).

Los días y el horario son la propuesta para este grupo y se confirman junto con la fecha de inicio, antes de abrir la matrícula. El calendario definitivo se entrega al confirmar la matrícula.

REQUISITOS

Lo que necesitas traer a la sesión 01.

No hay nivelación previa. La sesión 01 convierte llamadas síncronas del caso en eventos en la primera hora.

Programación

- ✓ Un lenguaje backend con soltura
- ✓ Concurrencia básica
- ✓ Git y pull requests

Datos

- ✓ SQL
- ✓ Transacciones en una base de datos
- ✓ Nociones de índices

Mensajería

- ✓ Haber usado alguna cola o broker
- ✓ HTTP y JSON

Herramientas

- ✓ Docker Compose a nivel de uso
- ✓ Terminal

Equipo y tiempo

- ✓ Computadora con 16 GB de memoria
- ✓ Seis horas de clase por semana
- ✓ Entre cuatro y seis horas de laboratorio
- ✓ 10 semanas de compromiso

Costos de herramientas que asumes tú `US$0 a 20 aprox.`

Estos costos NO están incluidos en el precio del programa. Todo el software del laboratorio es gratuito.

- El laboratorio corre en tu máquina con contenedores: Kafka, RabbitMQ, PostgreSQL, registro de esquemas y servicios del caso, todo de código abierto
- Si tu equipo no alcanza (se recomiendan 16 GB de memoria), una máquina virtual alquilada solo durante las prácticas cuesta del orden de US$15 en todo el programa
- No se usa ningún servicio gestionado de pago

PRECIO

60 horas para diseñar con hechos y dormir tranquilo cuando un proceso muere.

US$360

Pago único · USD · Incluye impuestos

Todavía no hay fecha de inicio. Escríbenos y te avisamos cuando la confirmemos: no cobramos nada antes de abrir la matrícula.

Incluye

- ✓ 20 sesiones en vivo por Zoom (60 horas lectivas)
- ✓ Grabación de todas las clases con acceso por 12 meses
- ✓ Sistema del caso con brokers, almacén de eventos y servicios listo para levantar
- ✓ Arnés para matar procesos en puntos elegidos del flujo
- ✓ Historia simulada de dos años de eventos para los laboratorios de versionado y reconstrucción
- ✓ Los 3 certificados progresivos
- ✓ Canal privado del grupo para dudas entre sesiones
- ✓ Actualizaciones del material por 12 meses

No incluye

- — Una máquina virtual alquilada si tu equipo no alcanza (rango de costos propios: US$0 a 20)
- — Exámenes ni certificaciones de terceros: este programa no prepara para ninguno
- — Servicios gestionados de mensajería: todo el laboratorio corre en local

Métodos de pago

Tarjeta de crédito · Tarjeta de débito internacional · PayPal · Apple Pay · Google Pay

Garantía de 14 días. Si dentro de los primeros 14 días del programa decides que no es para ti, te devolvemos el 100 % de lo pagado. No pedimos justificación.

Paddle actúa como comerciante registrado (merchant of record): emite el comprobante y gestiona los impuestos de cada país.

PREGUNTAS FRECUENTES

Preguntas frecuentes sobre el programa.

¿Las clases son en vivo o grabadas?

En vivo por Zoom: 20 sesiones de 3 horas, 2 veces por semana. Todas quedan grabadas y disponibles por 12 meses.

¿Cuándo empiezan las clases?

La fecha de inicio y el calendario definitivo todavía no están confirmados: se publican en esta misma ficha antes de que se abra la matrícula. Los días propuestos son Martes y jueves en horario de GMT-5 (Perú, Colombia, Ecuador y Panamá; en Ciudad de México es una hora menos). Si quieres que te avisemos cuando se confirmen, escribe a soporte@reactorcourses.com.

¿Qué nivel necesito?

Un lenguaje backend con soltura, SQL con transacciones y haber usado alguna cola o broker. No hace falta haber usado Kafka ni event sourcing antes.

¿Necesito una cuenta de nube?

No. Todo corre en tu máquina con contenedores y software libre. Se recomienda un equipo con 16 GB de memoria; si no alcanza, el rango de costos propios es de US$0 a 20, por alquilar una máquina virtual solo durante las prácticas.

¿El broker procesa cada mensaje exactamente una vez?

Se enseña exactamente hasta dónde llega. Kafka ofrece exactamente una vez al leer, procesar y escribir dentro de sus propios temas; cuando la salida va a otro sistema, la garantía depende de cómo coordines la posición del consumidor con lo que guardas. El programa construye esa parte con bandeja de salida y consumidores idempotentes, y la prueba matando procesos.

Ya cursé un programa de sistemas distribuidos o de plataforma de datos. ¿Me sirve este?

Sí, y conviene que sepas qué se repite. Cerca de un tercio de las sesiones toca problemas que esos programas también tratan: elegir entre log y cola, idempotencia y bandeja de salida, esquemas y su compatibilidad, sagas, mensajes fallidos y eventos publicados. Aquí el centro es modelar un dominio de negocio con eventos y operarlo, con otros laboratorios. Lo nuevo serían event storming, contextos y dueños de eventos, event sourcing, CQRS, versionado de eventos guardados, consistencia percibida y pruebas de sistemas asíncronos. Para reducir esa repetición el programa se acortó a 60 horas. No hay descuento ni convalidación entre programas.

¿Por qué PostgreSQL como almacén de eventos y no un producto especializado?

Porque permite ver cómo funciona por dentro un almacén de eventos y porque es software libre. Los almacenes especializados más conocidos hoy tienen licencias que no son de código abierto, y el programa no depende de ninguno.

¿Se enseña procesamiento de flujos en tiempo real?

No como tema propio. Las ventanas y agregaciones para análisis son el terreno de una plataforma de datos; aquí los eventos son el modelo del negocio transaccional.

¿Qué pasa si falto a una clase?

La grabación queda disponible y cada laboratorio tiene su punto de partida en el repositorio. El sistema se construye por capas: faltar a varias sesiones seguidas sin recuperarlas te deja sin piezas del proyecto final.

¿Los certificados tienen validez oficial?

No. Son certificados de Reactor Courses que acreditan las horas y competencias del programa. No son títulos universitarios ni certificaciones de terceros.

¿Cuánto tiempo debo dedicarle por semana?

Seis horas de clase más entre cuatro y seis horas de laboratorio. Considera entre 10 y 12 horas semanales durante 10 semanas.

¿Qué métodos de pago aceptan?

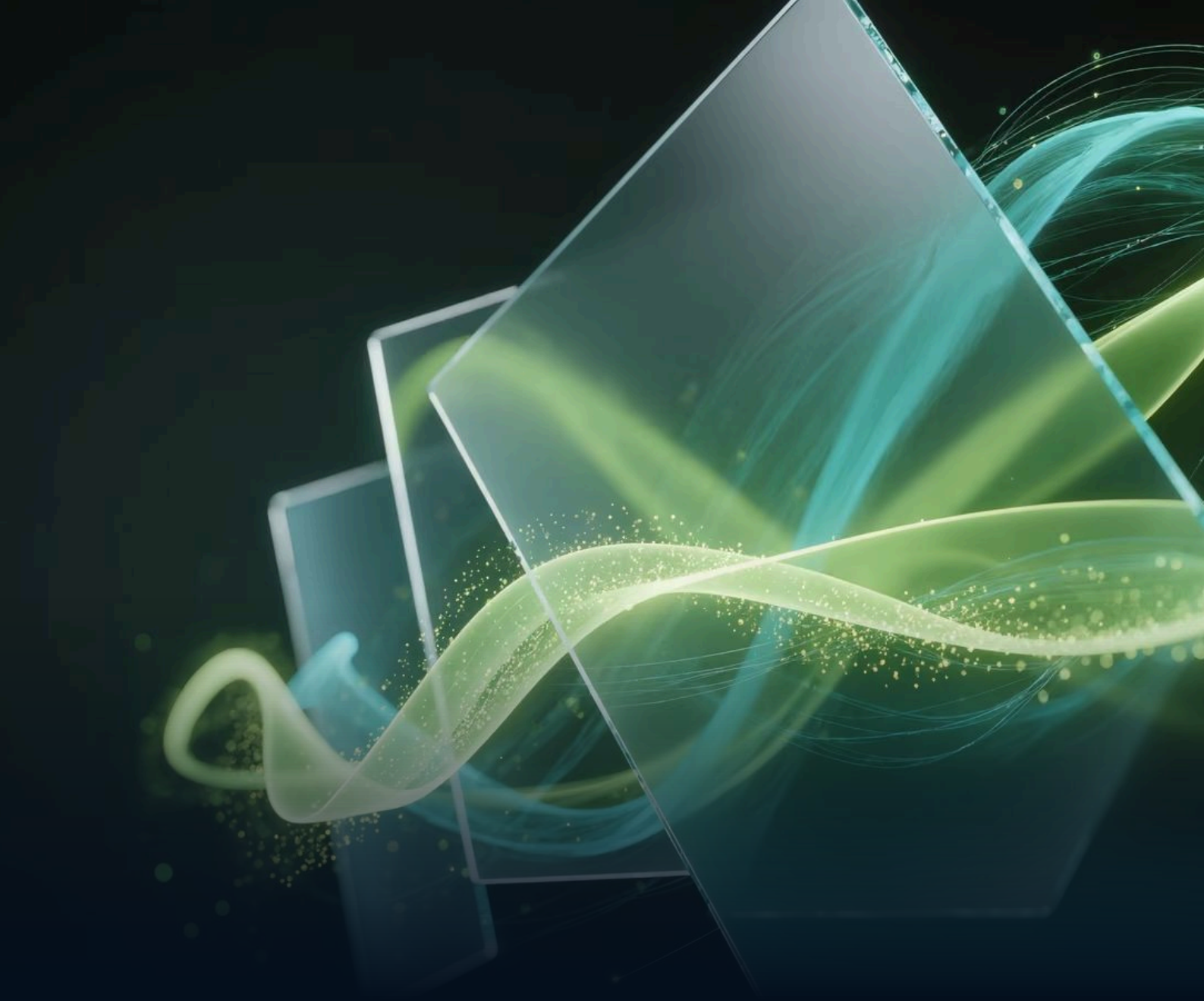
El pago lo procesa Paddle y acepta Tarjeta de crédito, Tarjeta de débito internacional, PayPal, Apple Pay, Google Pay. El cobro es en dólares. Importante: no procesamos Yape, Plin, OXXO, Mercado Pago ni transferencias locales. Si tu banco rechaza compras internacionales, habilítalas antes de pagar o usa PayPal.

¿Emiten factura o comprobante?

Sí. Paddle actúa como comerciante registrado (merchant of record): emite el comprobante y gestiona los impuestos de cada país. Lo recibirás por correo automáticamente al completar el pago.

¿Cuál es la política de reembolso?

Tienes 14 días desde el inicio del programa para pedir el reembolso completo escribiendo a soporte@reactorcourses.com. Publicamos el temario completo y los requisitos justamente para que decidas informado.



Reactor
COURSES

Ingeniería de software en la era de los agentes

¿Te quedó alguna duda?

Escríbenos a info@reactorcourses.com. La ficha al día, con el temario interactivo y la inscripción, está en reactorcourses.com/programas/event-driven-architecture.

`reactorcourses.com/programas/event-driven-architecture`

Información vigente al 23 de noviembre de 2026. El precio, las fechas y el temario pueden cambiar: la versión al día es la de la web.

Reactor Courses no está afiliado a Microsoft, Amazon Web Services, OpenAI, Anthropic ni Google. Los nombres de productos se mencionan únicamente con fines descriptivos del contenido académico.