MONOGRÁFICO 4

Kumoss

Kumoss: de una necesidad de infraestructura a un cambio revisable

Qué necesidad cubre, cómo funciona y qué aporta al SGIA. Siete lecciones, mapas visuales, matriz normativa y veinte preguntas con sus respuestas.

José Manuel Hernández Agudo

Fundamentos del SGIA e ISO/IEC 42001
Itinerario avanzado · 30 de octubre de 2026

Código y documentación fijados a la revisión **21eb7866ac99**. Estudio didáctico con fuentes, demostración oficial, actividades y preguntas razonadas.

Índice

Presentación 3

Lección 1 · Qué necesidad cubre Kumoss y cómo está construido 5

Lección 2 · Generar: de la petición al plan, la revisión y la aplicación 9

Lección 3 · Drift e importación: qué cambia en código, estado e infraestructura 13

Lección 4 · Identidad, roles y configuración efectiva 17

Lección 5 · Dependencias, datos y artefactos: seguir el recorrido completo 21

Lección 6 · Sesiones, informes y límites de la observabilidad 25

Lección 7 · La demostración oficial y su lectura desde el SGIA 29

Fichas en blanco para el alumno 34

Matriz de correspondencia normativa 35

Preguntas y respuestas razonadas 37

Cierre del monográfico 61

Fuentes y versión del estudio 63

Presentación

La pregunta que guía el monográfico

¿Cómo pasa una necesidad de una aplicación a una propuesta técnica que se pueda comprender, revisar y, cuando corresponda, ejecutar? Kumoss permite estudiar esa transición en un proyecto real de código abierto publicado por InditexTech. Está dirigido a desarrolladores de aplicaciones e ingenieros de plataforma: la persona expresa lo que necesita y el sistema prepara el trabajo con el contexto del repositorio, las convenciones y las herramientas de infraestructura. [S01, S02, S04]

Infraestructura significa aquí los recursos informáticos que permiten ejecutar, almacenar y conectar servicios. Infraestructura como código, abreviada IaC, consiste en describir esos recursos mediante archivos que un motor interpreta. Kumoss incorpora modelos de lenguaje para interpretar peticiones, proponer modificaciones y explicar resultados. El motor técnico, las reglas del programa y las decisiones de las personas tienen funciones propias. Comprender esa distribución evita atribuir a la IA toda la operación.

Qué necesidad cubre y qué aprenderás

El hilo conductor procede de la demostración oficial: una aplicación necesita almacenar imágenes y disponer de una dirección IP de salida fija. Esa dirección es la que ven otros servicios cuando la aplicación se conecta a ellos. No se trata de una empresa inventada para el curso: se estudia la petición publicada y su recorrido, con el alcance que explica la propia guía. [S03]

Pregunta	Respuesta que desarrolla el monográfico
¿Qué hace?	Prepara cambios IaC y ofrece recorridos de generación, reconciliación e importación.
¿Cómo?	Coordina contexto, modelos, herramientas de archivos, validaciones, planes e informes.
¿Con qué?	Interfaz web, núcleo de coordinación, modelos configurables, motor IaC, Git y sistemas de datos.
¿Cuándo se utiliza?	Para preparar cambios, ajustar código a recursos observados o incorporar recursos existentes a su gestión.
¿De qué forma?	Mediante sesiones con repositorio, ruta IaC, proveedor, ámbito y un modo de operación.
¿Qué beneficio muestra su diseño?	Convierte una petición en artefactos y resultados examinables. Medir ahorro o reducción de incidentes exigiría otra evaluación.

Las siete lecciones avanzan desde el vocabulario y la arquitectura hasta una lectura razonada de la demostración. Cada una termina con una pregunta documental y una respuesta corta. Puedes estudiarlas sin instalar la aplicación ni utilizar una cuenta cloud.

Fuente, versión y alcance de las conclusiones

El análisis utiliza la revisión 21eb7866ac995f6d6f3159f2ea342e472893638c, consultada el 29 de octubre de 2026; el manifiesto del núcleo declara la versión 1.0.3. Las referencias S01-S23 remiten a las fuentes primarias del estudio, y un mismo identificador puede reunir varios archivos. Se ha realizado una revisión estática dirigida: leer código, documentación y definiciones de pruebas. No se han ejecutado Kumoss, modelos, pruebas ni operaciones cloud. [S23]

La publicación de InditexTech y la demostración documentada no permiten deducir un despliegue global en Inditex, sus aprobadores, ahorros o certificaciones. El beneficio observado es funcional: existe un recorrido que relaciona petición, código, comprobaciones y resultados. La eficacia en una implantación exige evidencias propias.

SGIA significa sistema de gestión de la inteligencia artificial: el conjunto organizativo de políticas, responsabilidades y procesos con el que se gobierna la IA. En este curso, Kumoss aporta un objeto concreto para aplicar ese razonamiento. Su uso puede apoyar el sistema de gestión, pero los informes del producto y el análisis del repositorio no demuestran por sí solos conformidad con ISO/IEC 42001.

Lección 1 · Qué necesidad cubre Kumoss y cómo está construido

Entiende primero el problema y el vocabulario; después identifica qué hace la IA y qué ejecutan los demás componentes.

La necesidad: convertir una intención en una propuesta técnica

Una aplicación necesita recursos informáticos: procesamiento, almacenamiento y conexiones. Pedir una modificación exige traducir una necesidad a una configuración y comprobar su relación con lo que ya existe. Cloud designa recursos y servicios informáticos suministrados desde una plataforma remota. Kumoss organiza parte de esa preparación. Una persona puede expresar su petición en lenguaje natural y obtener cambios en archivos, comprobaciones técnicas e informes para revisar. El interés está en conectar la intención con resultados examinables. [S02, S04]

La demostración publicada parte de una aplicación que necesita almacenar imágenes y utilizar una IP de salida fija. Es una necesidad comprensible antes de conocer los nombres de los recursos cloud. Kumoss debe interpretarla dentro de un repositorio y un entorno. El texto inicial no es una orden enviada directamente al proveedor: inicia un proceso de preparación. [S03, S05]

Está dirigido a desarrolladores e ingenieros de plataforma. El desarrollador expresa la necesidad; la plataforma aporta contexto, convenciones y servicios. El beneficio funcional es disponer de un recorrido para producir y explicar propuestas. El repositorio no permite calcular cuánto tiempo ahorra ni cuántos errores evita.

Vocabulario antes de seguir

Término	Significado para este estudio
IaC, infraestructura como código	Archivos que describen recursos y sus propiedades de manera versionable.
HCL	Lenguaje de configuración compatible con Terraform utilizado para describir esos recursos.
Git, repositorio y rama	Control de versiones, lugar del código e historia y línea de trabajo separada.
Plan	Resultado técnico que anticipa operaciones sobre recursos; obtenerlo es distinto de ejecutarlo.
Apply, aplicación	Operación del motor que ejecuta un plan.
Estado, state	Registro que relaciona recursos gestionados con su representación IaC.
Pull request, PR	Propuesta de incorporar cambios a otra rama del repositorio, con diferencias revisables.
Sesión y ronda	La sesión agrupa el trabajo y su contexto; una nueva petición abre una ronda.

Estas definiciones permiten leer los recorridos oficiales sin confundir archivos, estado y recursos desplegados. OpenTofu y Terraform son motores IaC: interpretan la configuración y realizan operaciones como validar, planificar, importar o aplicar. El servicio distribuido utiliza OpenTofu por defecto y contempla Terraform como opción. [S07, S08, S22]

Qué piezas intervienen

La interfaz React es la aplicación web que recoge peticiones y presenta progreso, informes y acciones. El núcleo Python, expuesto mediante FastAPI, recibe solicitudes y coordina los servicios. API significa interfaz de programación: el contrato por el que otros programas solicitan operaciones. La interfaz web utiliza esa API; su recorrido visual no agota todos los caminos posibles del servidor. [S04]

LiteLLM adapta las llamadas a los modelos configurados. Un modelo de lenguaje interpreta y produce contenido a partir del contexto recibido. Un agente es el uso de ese modelo dentro de un bucle con instrucciones y herramientas concretas: el programa ofrece funciones, recibe solicitudes del modelo, ejecuta las funciones y devuelve resultados al contexto. El modelo no ejecuta por sí solo los comandos del motor. [S09, S10]

Pieza	Trabajo que realiza	Resultado que ayuda a explicar
Interfaz y núcleo	Recoger contexto y coordinar sesiones	Qué se pidió y cómo progresa la operación
Modelos y herramientas	Interpretar, consultar archivos, proponer ediciones e informes	Cómo se prepara la propuesta
Servicio IaC	Ejecutar operaciones de OpenTofu o Terraform	Qué valida, planifica o aplica el motor
Git	Versionar cambios y tramitar propuestas de incorporación	Qué archivos cambiaron
Sistemas de datos	Conservar sesiones, artefactos, estado y trazas	Dónde permanece información del proceso

El núcleo contempla un papel de modelo principal y otro pequeño. Distintos nombres de agentes representan tareas; no demuestran modelos distintos para cada una, entrenamiento propio de Inditex ni ejecución paralela. Tampoco hay que confundir compatibilidad de configuración con calidad comprobada para cualquier modelo. [S09, S10]

Contexto, integración y límites

La sesión identifica repositorio, ruta IaC, proveedor y ámbito. La ruta IaC es la carpeta sobre la que trabaja el motor. El ámbito delimita el contexto de recursos de la operación. Deben ser correctos para que una petición se interprete sobre el objeto pretendido; su presencia no reduce automáticamente los privilegios de las credenciales cloud.

Algunos servicios auxiliares se conectan mediante OpenAPI, una especificación de contratos de API. El proyecto aporta servicios de referencia de ejecución IaC y notificación. La resolución externa de repositorios y la autorización externa necesitan la integración correspondiente cuando se habilitan. Un

contrato describe cómo conectarse; no acredita que exista una instalación organizativa. [S04, S21]

La arquitectura permite localizar funciones y dependencias antes de valorar sus riesgos. PostgreSQL conserva información de aplicación; el almacenamiento de objetos, artefactos; Phoenix, trazas y versiones de instrucciones. Sus responsabilidades se desarrollan en las siguientes lecciones. La hoja de ruta trata MCP y A2A como integraciones previstas; no sustentan una integración actual con otros proyectos del curso. [S08, S23]

Comprueba tu comprensión

Pregunta documental: ¿Por qué una petición en lenguaje natural no equivale todavía a un cambio ejecutado?

Respuesta corta: Porque primero se interpreta y se transforma en código y comprobaciones. El efecto de aplicar corresponde al motor IaC, dentro del recorrido y las condiciones del modo utilizado. [S05, S17]

Amplía este punto en el curso

Módulo 4 · Inventario y alcance. Permite situar repositorio, ámbito, servicio y dependencias dentro del sistema que se gobierna.

Módulo 9 · Proveedores y ciclo de vida. Amplía el análisis de los servicios y componentes de los que depende la solución.

Fuentes de la lección

S01 · Organización oficial InditexTech

S02 · Presentación del proyecto

S03 · Demostración publicada y condiciones de la grabación

S04 · Arquitectura y finalidad

S05 · Cómo se formula y tramita una petición

S07 · Vocabulario oficial

S08 · Datos, estado y espacios de trabajo

S09 · Papeles de modelo y requisitos

S10 · Orquestación de modelos y herramientas

S21 · Operación y despliegue

S22 · Licencia, avisos y motor opcional

S23 · Historial de cambios y versión

El registro completo de archivos aparece al final del documento y en las fuentes del paquete.

Lección 1 · Mapa visual

KUMOSS · INFRAESTRUCTURA CON IA

Qué hace Kumoss y cómo está construido

Sitúa el papel de los agentes, el código de infraestructura y los servicios que sostienen la aplicación.

ANTES DE EMPEZAR

IaC = infraestructura como código: archivos que describen recursos. Un plan anticipa operaciones; todavía no las ejecuta.

01

Necesidad de la aplicación

Una persona pide recursos: almacenamiento, red o capacidad para que su aplicación funcione.

02

IA y coordinación

React recoge la petición; FastAPI coordina la sesión; LiteLLM conecta modelos que proponen código.

03

Código y motor IaC

HCL describe recursos. OpenTofu o Terraform validan, planifican y ejecutan según el modo elegido.

04

Resultados y servicios

Git versiona código; bases de datos, objetos y Phoenix conservan datos, artefactos y trazas.

CLAVE PARA INTERPRETAR

La petición no actúa sobre cloud por sí sola. El efecto depende del modo, el motor, el estado y las credenciales.

Fuente: InditexTech/kumoss · arquitectura, petición y componentes
Rev. 21eb7866ac995f6d6f3159f2ea342e472893638c
José Manuel Hernández Agudo · 30-10-2026 · Lectura estática

Lección 2 · Generar: de la petición al plan, la revisión y la aplicación

Sigue el recorrido que puede modificar recursos y distingue qué prepara el modelo, qué comprueba el motor y qué decide la organización.

Cuándo se utiliza Generar

Generar es el modo para preparar cambios solicitados sobre la infraestructura descrita en un repositorio. Puede crear, modificar o eliminar recursos según la petición y el contexto. Su resultado incluye una propuesta de código, comprobaciones e informes; cuando el recorrido llega a la fijación del plan, permite una aplicación posterior. Esta lección sigue ese modo: drift e importación tienen efectos diferentes. [S05, S06]

La persona identifica repositorio, proveedor, ámbito y ruta IaC, y expresa la necesidad. La documentación admite peticiones incompletas: el contexto y las convenciones pueden aportar valores, y el resultado debe explicar sus suposiciones. Esto hace útil revisar qué interpretó el sistema, incluso si la solicitud parecía clara. No hay una lista universal de parámetros obligatorios que el alumno deba inventar. [S05, S12]

Cómo se prepara el código

El programa coordina tareas de interpretación, selección de instrucciones y generación. Las convenciones son reglas que orientan nombres, recursos y configuraciones. Un prompt es el conjunto de instrucciones y contexto presentado al modelo. Kumoss combina plantillas locales con versiones de instrucciones obtenidas de Phoenix; conocer solo los archivos de reglas del repositorio no identifica necesariamente las versiones activas en una instalación. [S10, S14]

Las herramientas de archivos permiten al generador leer, escribir, sustituir y borrar contenido dentro del espacio de trabajo previsto. El programa ejecuta esas funciones. Después, el motor valida y planifica; si devuelve errores, el sistema puede alimentar otra tentativa de generación con esa información. El número de iteraciones está acotado. Un bucle de corrección facilita trabajar con errores, pero no garantiza que cualquier petición termine correctamente. [S10, S15]

La rama de sesión puede recibir commits y envíos de cambios antes de validar cada tentativa. Un commit es una versión registrada en Git. Por tanto, «todavía no se ha aplicado infraestructura» no significa que no haya escrituras persistentes: ya puede haber código y artefactos conservados. La incorporación de la rama al destino principal es otro hito.

Qué comprueban el motor y la revisión de cumplimiento

Un plan es la representación técnica de las operaciones previstas. Lo calcula OpenTofu o Terraform; el modelo puede ayudar a seleccionar objetivos o explicar el resultado, pero no sustituye ese cálculo. Generar contiene además una precomprobación de diferencias con lo desplegado, con su propio recorrido acotado. No debe convertirse su existencia en una promesa de reconciliación completa. [S15, S17]

Cuando está habilitada, la revisión de cumplimiento presenta al modelo la petición inicial y el plan. El modelo devuelve hallazgos y el programa calcula un resultado a partir de sus severidades. Esa regla programada no garantiza que el modelo detecte todas las infracciones. El informe corresponde a los criterios configurados; no certifica cumplimiento legal universal ni aceptabilidad de todo riesgo. [S11]

YAML es el formato del archivo de configuración. El YAML distribuido activa revisión de cumplimiento y bloqueo por impacto alto; los valores de clase son falsos cuando faltan esas opciones. La configuración efectiva determina qué controles intervienen. Una sesión bloqueada puede consultarse y abrir una PR para revisión; integrar y aplicar quedan impedidos mientras el bloqueo corresponda. [S16, S20]

Revisar, incorporar y aplicar son hitos diferentes

La PR permite examinar diferencias en el proveedor Git. La interfaz ofrece un recorrido de revisión y una confirmación de comprensión antes de incorporar y aplicar. La API admite aplicar sin imponer una PR previa, por lo que no se puede presentar esa secuencia visual como requisito técnico de todos los caminos. [S12, S13, S16]

Hito	Qué permite conocer	Qué debe interpretarse por separado
Código y plan	Archivos propuestos y operaciones previstas	Adecuación de la intención y del alcance
Informe	Explicación y comprobaciones configuradas	Límites de detección y criterio aplicable
Incorporación en Git	Integración de los cambios	Autorización y ejecución sobre recursos
Resultado de apply	Lo que informa el intento del motor	Estado posterior y aceptación del resultado

AUT significa aquí autoridad de cambio designada por la organización. Sus atribuciones y procedimiento determinan qué autorización corresponde; un botón y un informe favorable no identifican por sí solos esa autoridad. Tampoco se presume una firma universal o una doble aprobación obligatoria. Si el procedimiento permite decisiones bajo una política previa, habría que comprobar sus condiciones y la evidencia por cambio. El código revisado no acredita un motor nativo de automerge ni de autorización delegada.

Por qué importa el plan fijado

Plan fijado significa conservar el plan y la raíz de trabajo preparados para aplicarlos sin regenerar el plan en ese momento. El controlador exige ese artefacto y lo descarta al terminar el intento, incluso ante un fallo. El beneficio es vincular la aplicación al resultado preparado; la comprobación técnica no demuestra quién lo revisó o autorizó. [S07, S17]

La guía contempla éxito, aplicación parcial y fallo. No documenta reintento ni reversión automáticos de lo aplicado. Otro intento requiere preparar un nuevo plan e interpretar qué quedó de la operación anterior. Conservar esa distinción conecta propuesta, decisión y resultado sin atribuir al producto la aceptación organizativa.

Comprueba tu comprensión

Pregunta documental: ¿Obtener un plan válido demuestra que el cambio ya se ejecutó?

Respuesta corta: No. El plan anticipa operaciones; apply es otra operación, exige el plan fijado y tiene un resultado propio que debe leerse. [S16, S17]

Amplía este punto en el curso

Módulo 8 · Supervisión humana. Explica la diferencia entre disponer de información técnica y ejercer una decisión humana con autoridad.

Módulo 6 · Tratamiento y controles. Relaciona las condiciones de aceptación y los controles con el riesgo que se pretende tratar.

Fuentes de la lección

S05 · Cómo se formula y tramita una petición

S06 · Comparación de los modos

S07 · Vocabulario oficial

S10 · Orquestación de modelos y herramientas

S11 · Revisión de cumplimiento y cálculo de su resultado

S12 · Preguntas frecuentes y límites del recorrido

S13 · Roles, propiedad de sesiones y acceso

S14 · Composición de instrucciones

S15 · Generación, validación y fijación

S16 · Revisión y acciones de la interfaz

S17 · Aplicación del plan existente

S20 · Configuración distribuida y valores de clase

El registro completo de archivos aparece al final del documento y en las fuentes del paquete.

Lección 2 · Mapa visual

KUMOSS · INFRAESTRUCTURA CON IA

Generar: revisión y aplicación del plan fijado

Sigue el recorrido que puede modificar la infraestructura y distingue sus hitos de decisión.

ANTES DE EMPEZAR

Plan = operaciones previstas. Apply = ejecutarlas. Plan fijado = artefacto conservado para aplicar sin regenerarlo.

01

Del texto al código

Petición y contexto del repositorio → los agentes proponen cambios HCL. Proponer no es autorizar.

02

Validar y planificar

El motor comprueba el código y obtiene un plan. Los errores permiten iterar dentro de los límites configurados.

03

Revisar y decidir

Los informes apoyan la decisión. Una pull request propone integrar el código en Git; es una acción separada.

04

Aplicar explícitamente

Apply ejecuta el plan fijado. Cada intento consume ese plan; un fallo no revierte automáticamente lo aplicado.

CLAVE PARA INTERPRETAR

La AUT (autoridad de cambio) decide manualmente o con política previa comprobada por cambio. Kumoss no acredita un automerge nativo.

Fuente: InditexTech/kumoss · generación, revisión y aplicación
Rev. 21eb7866ac995f6d6f3159f2ea342e472893638c
José Manuel Hernández Agudo · 30-10-2026 · Lectura estática

Lección 3 · Drift e importación: qué cambia en código, estado e infraestructura

Elige el modo por la necesidad y por sus efectos: reconciliar lo observado e incorporar recursos existentes son operaciones distintas.

Tres objetos que conviene separar

El código IaC describe recursos; el estado del motor relaciona esa descripción con recursos gestionados; la infraestructura observada es lo que encuentra la herramienta en el entorno. Pueden existir diferencias entre ellos. Mantener esa distinción permite entender qué pretende cada modo y en qué momento se producen efectos. No basta con preguntar si aparece el botón apply. [S06, S08]

Drift significa diferencia entre la infraestructura observada y la que describe el código. Importar significa incorporar recursos existentes a su gestión mediante IaC. Ambos recorridos pueden modificar código y terminar con una propuesta de incorporación al repositorio; la importación, además, escribe estado durante la ronda. El alcance y la revisión deben considerar esos efectos antes de empezar.

Cuándo utilizar drift y hacia dónde reconcilia

El modo dedicado de drift prepara modificaciones de HCL para adoptar en el código lo encontrado desplegado. La modalidad parcial trabaja con objetivos seleccionados; la completa se extiende a toda la raíz IaC. Raíz IaC es la carpeta de configuración sobre la que se ejecuta el motor. «Completo» se refiere a esa raíz y a su recorrido, no a una auditoría exhaustiva de todos los recursos de una cuenta cloud. [S06]

La orientación es esencial: Kumoss propone cambiar código hacia lo observado. El recorrido dedicado no restaura la nube a la versión anterior del código, no fija un plan para aplicar y no llama a apply. La reconciliación puede ser útil para documentar diferencias, pero la decisión de aceptar lo observado sigue abierta. La fuente técnica de comparación y la referencia organizativa de lo autorizado cumplen funciones distintas.

La selección de objetivos, la descomposición de tareas y parte del tratamiento de excepciones utilizan modelos. El programa aplica las respuestas y los límites del bucle. Se puede terminar con diferencias pendientes o con operaciones excluidas. Es necesario examinar informe y cambios, en lugar de interpretar la palabra «reconciliado» como garantía universal.

Qué sucede al importar

Importar resulta pertinente cuando existen recursos que todavía no figuran como gestionados en el estado IaC correspondiente. El recorrido compara inventario del ámbito con el estado, aplica exclusiones y prepara candidatos. Inventario significa aquí el conjunto de recursos devuelto por los mecanismos de descubrimiento disponibles; su alcance depende de credenciales, proveedor, filtros y límites. [S18]

En importación parcial, una petición ayuda a seleccionar recursos. La completa intenta incorporar los candidatos que queden después de las exclusiones previstas. Se preparan bloques HCL y se resuelven direcciones del código e identificadores de recursos para las operaciones de importación. El motor

escribe el estado durante la ronda, antes de revisar una PR posterior.

La versión estudiada implementa descubrimiento e importación para AWS, Azure y GCP. No debe extenderse esa cobertura a todos los proveedores que puedan aparecer en otras partes de la aplicación. Tampoco «todos los candidatos» equivale a todos los recursos de la cuenta: el descubrimiento tiene requisitos y filtros propios. [S06, S18]

Si unas importaciones tienen éxito y algo posterior falla, esas escrituras logradas pueden permanecer. La limpieza de archivos temporales no deshace automáticamente el estado. La lectura estática también distingue la generación y planificación inicial de los reintentos de importación: no son un único bucle que repita siempre todos los pasos. Su resultado requiere examinar intentos, omisiones y diagnósticos.

Comparar efectos antes de decidir

Familia	Necesidad	Efectos relevantes	Final del recorrido documentado
Generar	Preparar el cambio solicitado	Código, artefactos y plan; recursos y estado al aplicar	Puede continuar a apply del plan fijado
Drift parcial o completo	Reflejar diferencias observadas en el código	Código, rama e informes	Incorporación al repositorio, sin apply posterior
Importación parcial o completa	Gestionar recursos existentes con IaC	Código y estado durante la ronda	Incorporación al repositorio, sin el apply posterior de Generar

La revisión de cumplimiento y su bloqueo asociado se documentan para Generar cuando están habilitados. No hay base para atribuir las mismas comprobaciones a todos los modos. Los roles mínimos y las opciones de interfaz también se distinguen; compartir sesión o pantalla no convierte los recorridos en equivalentes. [S06, S13]

Algunos comentarios antiguos del código mencionan drift junto a la aplicación. El análisis sigue los controladores y la tabla funcional: solo Generar fija el plan en el recorrido examinado. Un comentario aislado no acredita una capacidad adicional.

Beneficio y pregunta de gobierno

La utilidad funcional de drift es preparar una representación revisable de diferencias. La de importar es relacionar recursos existentes con código y estado gestionado. En ambos casos, el SGIA necesita determinar ámbito, responsable y criterio de aceptación, además de qué efectos se autorizan antes de la ronda. Revisar una PR al final no antecede a las escrituras de estado que la importación ya haya realizado.

Comprueba tu comprensión

Pregunta documental: ¿Por qué «Importar no tiene un apply posterior» no significa «Importar es de solo lectura»?

Respuesta corta: Porque ejecuta operaciones de importación que escriben el estado durante la ronda, antes de la revisión posterior de la PR. [S18]

Amplía este punto en el curso

Módulo 4 · Inventario y alcance. Ayuda a distinguir el sistema y las versiones que se están examinando.

Módulo 5 · Riesgos e impactos. Permite valorar una diferencia antes de convertirla en una condición aceptada.

Fuentes de la lección

S06 · Comparación de los modos

S08 · Datos, estado y espacios de trabajo

S13 · Roles, propiedad de sesiones y acceso

S15 · Generación, validación y fijación

S17 · Aplicación del plan existente

S18 · Recorrido de importación y escritura de estado

El registro completo de archivos aparece al final del documento y en las fuentes del paquete.

Lección 3 · Mapa visual

KUMOSS · INFRAESTRUCTURA CON IA

Drift e importación: código, estado y autorización

Diferencia lo desplegado, lo representado en código y lo que la organización ha autorizado.

ANTES DE EMPEZAR

Drift = diferencia entre lo observado y el código. Estado IaC = registro que vincula recursos y su representación.

01

Tres referencias distintas

Infraestructura observada, código y estado autorizado pueden diferir. Encontrar algo no significa aprobarlo.

02

Reconciliar drift

Ajusta código hacia lo desplegado. Drift parcial o completo termina en Git: no hay apply posterior.

03

Importar recursos

Incorpora recursos existentes al código y escribe estado DURANTE la ronda. Importar no es solo leer.

04

Efectos según el modo

Generar prepara un plan aplicable; drift actualiza código; importación actualiza código y estado.

CLAVE PARA INTERPRETAR

No traslades los controles de Generar a todos los modos. La revisión de cumplimiento y su bloqueo asociado tienen condiciones específicas.

Fuente: InditexTech/kumoss · modos operativos e importación
Rev. 21eb7866ac995f6d6f3159f2ea342e472893638c
José Manuel Hernández Agudo · 30-10-2026 · Lectura estática

Lección 4 · Identidad, roles y configuración efectiva

Relaciona quién es la persona, qué operaciones permite la aplicación y qué condiciones están realmente activadas.

Tres preguntas distintas sobre el acceso

Autenticación significa comprobar la identidad de quien se conecta. Autorización significa decidir qué puede hacer esa identidad sobre un objeto. Autoridad organizativa significa tener atribuciones para tomar una decisión dentro del procedimiento aplicable. Kumoss incorpora mecanismos técnicos de las dos primeras; su correspondencia con la tercera necesita definirse en la organización. [S13]

También hay que separar permisos de la aplicación y permisos cloud. Un rol de Kumoss puede permitir iniciar una operación, mientras las credenciales del servicio IaC determinan qué acciones podrá intentar el motor en el proveedor. El ámbito escrito en una petición no acredita que esas credenciales estén limitadas exactamente a él.

Esta separación sirve antes del uso y al revisar cambios: permite preguntar quién inicia, quién consulta, quién desbloquea y con qué privilegios actúan los servicios. La respuesta debe describir condiciones comprobables, no deducirse del nombre de un botón.

Cómo se reconoce a una persona

OIDC, OpenID Connect, es un protocolo para recibir una identidad verificada por un proveedor externo. Cuando está habilitado con un emisor configurado, el núcleo requiere y valida un token: comprueba, entre otras condiciones, emisor, destinatario y vigencia. Un token es la credencial digital presentada en esa conexión. La identidad validada se relaciona con el usuario y los roles de la aplicación. [S13, S21]

El YAML distribuido deja vacío el emisor OIDC. En esa situación, las peticiones utilizan una identidad local de desarrollo con roles elevados: devops y administrador del panel. Esta descripción corresponde al archivo publicado, no a una instalación observada en Inditex. Cambiar el indicador de entorno a production no activa por sí solo las barreras de seguridad que requiere un despliegue. [S20]

Por ello, una revisión debe identificar qué archivo y variables de entorno se cargan. Configuración efectiva significa los valores realmente utilizados al arrancar y operar, después de considerar esas fuentes. Documentar la capacidad OIDC sin comprobar si está activada deja incompleta la explicación del acceso.

Roles y propiedad de las sesiones

El proyecto utiliza dos jerarquías. Los roles operativos developer y devops habilitan operaciones; los roles del panel viewer, editor y admin habilitan consultas y administración. Son nombres técnicos del producto, no cargos empresariales acreditados. [S13]

Mecanismo	Capacidad observada	Pregunta que sigue abierta en la organización
Developer	Generar, importar y aplicar, con las demás condiciones	¿A quién corresponde recibirlo y para qué alcance?
Devops	Añade el acceso a drift	¿Qué responsabilidades justifican esa capacidad?
Viewer del panel	Consultar sesiones de otros usuarios	¿Qué información puede examinar legítimamente?
Editor del panel	Bloquear o desbloquear sesiones	¿Qué criterio debe aplicar y registrar?
Admin del panel	Gestionar roles	¿Cómo se revisan altas, cambios y bajas?

La propiedad de sesión es otra condición: continuar una sesión, integrar su PR y aplicar requieren que quien actúa sea su propietario, además de los demás controles. El editor del panel puede bloquear o desbloquear por la ruta administrativa sin ser propietario. Administrar o consultar una sesión no permite automáticamente todas sus acciones. El listado ordinario se limita al usuario; el panel amplía la lectura según su rol.

El recorrido web ofrece confirmaciones de revisión. El servidor examinado no acredita una segunda persona obligatoria para aprobar cada aplicación. Las protecciones de rama y revisores exigidos por Git podrían añadir controles, pero hay que comprobarlos en el proveedor utilizado.

Autorización externa: dónde interviene

Authz es la abreviatura empleada para el servicio de autorización externa. El asistente web puede consultarlo antes de continuar con la preparación inicial; esa comprobación previa se denomina preflight. En el YAML distribuido está desactivado. Si no está configurado, ese servicio no añade una denegación. Si se habilita y falla la comunicación, la ruta devuelve error. [S13, S20]

Las rutas revisadas de generación, aplicación e incorporación de PR comprueban identidad, roles y propiedad, sin repetir esa consulta externa de proyecto. Generar recalcula el bloqueo de la ronda; aplicar e integrar comprueban el bloqueo antes de operar. Por tanto, no se puede describir el preflight opcional de la web como una autorización de proyecto obligatoria en cada operación de API. El beneficio de una integración externa depende de dónde se invoque y cómo se utilice su resultado.

Qué interruptores cambian el recorrido

El archivo distribuido activa revisión de cumplimiento y bloqueo por impacto alto; los valores de clase son false si esas opciones faltan. También activa notificaciones, aunque su valor de clase sin configuración es desactivado. La resolución externa de repositorios y la autorización externa aparecen desactivadas. Esta diferencia enseña por qué «valor por defecto» requiere indicar de qué fuente se habla. [S20]

Los bloqueos tienen efecto de servidor sobre integrar y aplicar. Un editor puede modificarlos; una nueva ronda de Generar los recalcula. Esas capacidades apoyan el control operacional, pero una decisión de desbloqueo necesita interpretarse con el criterio aplicable y la evidencia conservada. No se impone una

firma universal ni se presume que un cambio de booleano pruebe todo el proceso de autorización.

Comprueba tu comprensión

Pregunta documental: ¿La consulta externa de autorización es obligatoria en cada apply de la API examinada?

Respuesta corta: No. Es una comprobación previa opcional del asistente web; apply aplica otros controles del servidor sin repetir esa consulta externa. [S13]

Amplía este punto en el curso

Módulo 3 · Política, liderazgo y responsabilidades. Sitúa las atribuciones de decisión y su relación con las funciones operativas.

Módulo 7 · Datos y seguridad. Amplía identidad, privilegios y protección de la información.

Fuentes de la lección

S13 · Roles, propiedad de sesiones y acceso

S16 · Revisión y acciones de la interfaz

S17 · Aplicación del plan existente

S20 · Configuración distribuida y valores de clase

S21 · Operación y despliegue

El registro completo de archivos aparece al final del documento y en las fuentes del paquete.

Lección 4 · Mapa visual

KUMOSS · INFRAESTRUCTURA CON IA

Identidad, roles y configuración efectiva

Examina quién puede actuar y por qué los valores de desarrollo requieren una lectura precisa.

ANTES DE EMPEZAR

OIDC identifica al usuario mediante un proveedor de identidad. El rol habilita acciones; la propiedad limita la sesión.

01

Identidad local u OIDC

Sin emisor OIDC, el YAML distribuido usa identidad local con devops y panel admin. No describe un despliegue real.

02

Operaciones en la API

Developer genera, importa y aplica; devops añade drift. Continuar, integrar y aplicar exigen propiedad de sesión.

03

Panel e interfaz web

Viewer lee; editor bloquea y desbloquea; admin gestiona roles. La web reserva importar a devops.

04

Autorización externa

Es una comprobación previa opcional del asistente web. No se repite en todas las rutas operativas de la API.

CLAVE PARA INTERPRETAR

Configuración efectiva: el YAML activa cumplimiento y bloqueo por impacto alto; los valores de clase son false. Comprueba qué carga cada instalación.

Fuente: InditexTech/kumoss · roles, acceso y configuración
Rev. 21eb7866ac995f6d6f3159f2ea342e472893638c
José Manuel Hernández Agudo · 30-10-2026 · Lectura estática

Lección 5 · Dependencias, datos y artefactos: seguir el recorrido completo

Identifica qué información necesita la solución, dónde se conserva y de qué componentes depende cada resultado.

La solución comprende más que un modelo

Una dependencia es un componente o servicio del que otro necesita para funcionar. Kumoss depende de modelos, bibliotecas, motor IaC, proveedor Git, almacenamiento y servicios auxiliares. El núcleo coordina ese conjunto. Identificarlo permite asignar mantenimiento, acceso y condiciones de uso; el nombre del modelo por sí solo no describe el sistema. [S04, S09]

Un artefacto es un resultado conservado del trabajo, como un informe, un plan o una copia de cambios. La utilidad de esos resultados depende de relacionarlos con sesión, ronda y contexto. No todos se guardan en el mismo lugar ni ofrecen las mismas posibilidades de consulta y recuperación. [S08]

El beneficio de separar las piezas es concreto: si se quiere conocer qué cambió, se examina Git; para entender qué contempló el motor, se consulta el plan; para relacionar recursos gestionados, se necesita el estado. Sustituir uno por el resumen de otro puede ocultar información relevante.

Qué información llega a los modelos

Las entradas pueden incluir petición, historia de conversación, instrucciones, contenido leído del repositorio, diferencias de archivos, planes y errores. Las herramientas devuelven resultados al contexto del agente; por tanto, revisar solo el primer mensaje del usuario no describe todo lo enviado. La revisión de cumplimiento recibe la petición inicial y el plan. [S10, S11, S12]

El proveedor y modelo concretos dependen de configuración. Los papeles principal y pequeño distribuyen tareas; no acreditan entrenamiento propio con datos de una empresa. El uso de un servicio debe examinarse con sus condiciones, destinos y configuración reales. Esta revisión no ha enviado datos ni inspeccionado contratos o regiones de una implantación.

La herramienta de búsqueda puede consultar un motor, leer páginas y resumir contenido con otro paso de IA. Ese contenido ayuda a preparar respuestas, pero incorpora fuentes y dependencias adicionales. Una explicación de coste obtenida así debe interpretarse como estimación, no como facturación realizada. [S10, S12]

Dónde permanece la información

Backend significa aquí el sistema donde el motor guarda el estado.

Componente	Qué conserva o utiliza	Utilidad y límite
Git	Código, ramas y versiones publicadas	Permite revisar cambios; incorporar una rama no demuestra el resultado cloud.

Componente	Qué conserva o utiliza	Utilidad y límite
PostgreSQL del núcleo	Usuarios, sesiones, rondas, estados y metadatos	Relaciona el proceso; no convierte todos sus registros en auditoría inmutable.
Almacenamiento de objetos	Informes, planes textuales y cambios conservados	Permite consultar artefactos mediante enlaces con sus condiciones de acceso.
Backend de estado IaC	Relación entre recursos gestionados y código	Es necesario para operaciones posteriores; debe conservarse y protegerse.
Volumen de trabajo	Archivos usados durante una ejecución y plan fijado	Su persistencia y limpieza afectan a la posibilidad de aplicar.
Phoenix	Trazas y versiones de instrucciones	Ayuda a investigar y a conocer reglas; añade un destino de datos.
Redis	Caché de información de sesión	Agiliza lecturas; la documentación no lo presenta como almacén durable del dominio.

El estado remoto es necesario porque los espacios de trabajo son temporales y el estado local no se conserva en Git. La configuración distribuida hace que Kumoss gestione el estado en un bucket separado del de artefactos. Un bucket es un contenedor lógico de objetos. Puede configurarse otro modelo en el que el repositorio declare su backend remoto. [S08, S20]

Cuando Kumoss lo gestiona, la identidad del estado relaciona repositorio, ámbito y ruta IaC; no es simplemente el identificador de la sesión. Varias sesiones pueden apuntar al mismo estado. Hay que comprobar esa correspondencia y la recuperación prevista al evaluar una instalación.

Dos persistencias que conviene distinguir

Los informes y planes textuales conservados no son el archivo de plan ejecutable fijado. Este último se conserva con la raíz IaC preparada en el volumen compartido para apply. La limpieza ordinaria del espacio temporal y el descarte del plan fijado son operaciones diferentes. Consultar o conservar el informe no reúne todo lo necesario para repetir una aplicación. [S08, S17]

Los enlaces de lectura a artefactos tienen una vigencia configurable, de 48 horas en la configuración distribuida. Compartir una sesión o copiar un enlace puede ampliar quién accede a su contenido durante esa vigencia. Su protección requiere considerar almacenamiento, permisos y conservación; esta revisión no ha probado revocación de enlaces ni accesos reales. [S20, S21]

Qué revisar en las dependencias

El proyecto publica licencia y avisos. La evaluación debe incluir las condiciones de los componentes elegidos, entre ellos Terraform opcional y los servicios externos. La licencia de Kumoss no sustituye las licencias o términos de toda la cadena. Tampoco enumerar dependencias demuestra que sean adecuadas o estén actualizadas en una instalación. [S22]

Para el SGIA, este mapa ayuda a preguntar quién mantiene modelos y motor, qué datos necesita cada destino, quién accede, cuánto se conserva y cómo se recupera. Las respuestas deben apoyarse en configuración y evidencias del uso previsto. El repositorio proporciona una base de estudio; no

documenta los acuerdos y decisiones de una organización concreta.

Comprueba tu comprensión

Pregunta documental: ¿El informe consultable y el estado IaC contienen la misma información y cumplen la misma función?

Respuesta corta: No. El informe explica resultados; el estado relaciona recursos gestionados y código y condiciona operaciones posteriores. Se conservan y protegen según funciones distintas. [S08]

Amplía este punto en el curso

Módulo 9 · Proveedores y ciclo de vida. Relaciona modelos, servicios y componentes con dependencias que se mantienen y revisan.

Módulo 7 · Datos y seguridad. Profundiza en acceso, conservación y protección de información y artefactos.

Fuentes de la lección

S04 · Arquitectura y finalidad

S08 · Datos, estado y espacios de trabajo

S09 · Papeles de modelo y requisitos

S10 · Orquestación de modelos y herramientas

S11 · Revisión de cumplimiento y cálculo de su resultado

S12 · Preguntas frecuentes y límites del recorrido

S17 · Aplicación del plan existente

S19 · Trazas y datos de la operación

S20 · Configuración distribuida y valores de clase

S21 · Operación y despliegue

S22 · Licencia, avisos y motor opcional

El registro completo de archivos aparece al final del documento y en las fuentes del paquete.

Lección 5 · Mapa visual

KUMOSS · INFRAESTRUCTURA CON IA

Dependencias, datos y artefactos

Localiza qué información maneja el sistema y qué servicios forman parte de su funcionamiento.

ANTES DE EMPEZAR

Artefacto = archivo producido por el proceso. Traza = registro detallado de pasos y llamadas de los agentes.

01

Entradas al modelo

La petición, el contexto del repositorio, los planes y los errores pueden llegar al modelo configurado.

02

Cadena de dependencias

LiteLLM conecta modelos; Git, motor IaC, búsqueda y notificaciones añaden servicios y condiciones.

03

Persistencia repartida

Git conserva código; PostgreSQL, datos de sesión; objetos, artefactos; backend IaC, estado.

04

Trazas y acceso

Phoenix registra entradas, salidas y herramientas. Hay que revisar acceso, conservación y datos enviados.

CLAVE PARA INTERPRETAR

No basta con ubicar el modelo: hay que seguir la información en todos sus destinos y revisar proveedores, licencias, versiones y conservación.

Fuente: InditexTech/kumoss · datos, modelos y operación
Rev. 21eb7866ac995f6d6f3159f2ea342e472893638c
José Manuel Hernández Agudo · 30-10-2026 · Lectura estática

Lección 6 · Sesiones, informes y límites de la observabilidad

Aprende a interpretar progreso, planes, resultados y trazas sin convertir una señal técnica en una conclusión más amplia.

Seguir una operación que tarda

Una sesión relaciona peticiones y resultados con su contexto. Cada ronda añade trabajo, historia y artefactos. Kumoss ejecuta operaciones de manera asíncrona: puede aceptar una petición y continuar su procesamiento después de responder. La interfaz presenta estados y progreso mientras el núcleo coordina el trabajo. «Aceptada» describe la recepción; todavía no acredita éxito. [S04, S08]

Esta distinción importa al utilizar el producto. La pregunta «¿ha terminado?» necesita concretar qué ha terminado: recibir la solicitud, generar código, calcular un plan, importar recursos o aplicar. Cada etapa puede producir información y efectos diferentes. Un resumen favorable de una fase no sustituye la lectura del resultado de la siguiente.

Observabilidad es la posibilidad de comprender qué ocurre dentro de un sistema a partir de las señales que ofrece. Kumoss conserva estados, informes, artefactos y trazas. Su utilidad consiste en dar contexto para seguir e investigar operaciones; su suficiencia depende de la pregunta que se quiera resolver.

Qué muestra un informe

Los informes explican propuestas y resultados con un formato comprensible. Pueden incluir resumen, impacto, estimación de coste y explicación de cambios o diferencias. El modelo utiliza información del proceso y herramientas disponibles para elaborar esa explicación. El informe ayuda a revisar, pero hay que relacionarlo con el código y las salidas técnicas que interpreta. [S10, S12, S16]

Una estimación mensual es una previsión elaborada bajo datos y supuestos; no es una factura ni un ahorro demostrado. Un informe de cumplimiento corresponde a reglas configuradas y hallazgos del modelo; no acredita que se hayan detectado todas las infracciones. Una descripción de lo aplicado debe contrastarse con el resultado del motor y, cuando corresponda, con el estado posterior del servicio. [S11]

Señal	Pregunta que ayuda a responder	Límite de la conclusión
Petición aceptada	¿Se recibió el trabajo?	No demuestra que haya terminado.
Código y diferencias	¿Qué archivos se proponen o publicaron?	No demuestran aplicación cloud.
Plan	¿Qué operaciones prevé el motor?	No demuestra ejecución.
Informe de aplicación	¿Qué resultado describe el intento?	Requiere interpretar fallos y efectos parciales.

Señal	Pregunta que ayuda a responder	Límite de la conclusión
Traza	¿Qué pasos, entradas y salidas quedaron registrados?	No demuestra por sí sola autorización o eficacia.

El beneficio de mantener estas señales relacionadas es reconstruir mejor el recorrido. También permite detectar qué evidencia falta antes de emitir una conclusión.

Resultado del motor y estado de la sesión

La guía de aplicación distingue éxito, aplicación parcial y fallo. Una aplicación parcial puede dejar efectos que necesitan investigarse. La eliminación del plan fijado al terminar el intento no revierte los recursos; la limpieza del directorio temporal tampoco deshace Git o estado. [S16, S17]

La revisión estática ha localizado una divergencia entre una expectativa de prueba y el controlador de aplicación: una rama que recibe un resultado de apply desfavorable guarda el resumen y notifica sin lanzar la excepción que espera esa prueba. El proceso que actualiza estados podría terminar marcando la ronda como completada. Es una lectura del código, no un fallo reproducido durante esta revisión. Enseña por qué conviene examinar el resultado técnico y su explicación, además de la etiqueta de estado.

Ante incertidumbre operativa, la pregunta correcta es qué acción llegó a realizarse y qué permanece pendiente. No se debe deducir una recuperación automática de que la interfaz deje de mostrar actividad. La guía de producción requiere planificar continuidad y recuperación; aquí no se han probado reinicios ni restauraciones. [S19, S21]

Qué aportan las trazas y qué datos contienen

Una traza relaciona pasos de una ejecución y sus metadatos. Phoenix recibe telemetría, es decir, información enviada para observar el proceso. El código puede registrar identificadores de sesión y usuario, proveedor cloud, repositorio, ruta, mensajes, llamadas a herramientas, parámetros, resultados y contadores de tokens. Un token es una unidad de texto utilizada por el modelo para procesar y contabilizar contenido. [S19]

Ese detalle ayuda a investigar cómo se llegó a una propuesta. También amplía el recorrido de la información: una revisión de acceso y conservación debe incluir las trazas, además de la conversación y el repositorio. El filtrado selectivo observado en la serialización no demuestra una eliminación general de secretos o datos personales. No se han inspeccionado trazas reales ni demostrado una fuga.

Trazabilidad funcional y evidencia de gestión

Trazabilidad significa poder relacionar elementos del proceso: petición, contexto, artefactos y resultado. El almacenamiento de información con fechas facilita esa relación. No basta para denominarla registro inmutable o prueba de no repudio. En los caminos revisados de desbloqueo y modificación de roles no se observa el registro completo de actor, motivo y valores anteriores y posteriores.

La organización necesita definir qué decisiones documenta y cómo protege esa información. Si un procedimiento exige una evidencia específica de autorización, debe comprobarse esa condición sin convertirla en requisito universal. La presencia o ausencia de un campo en el producto no demuestra por sí sola conformidad o incumplimiento de una empresa.

Comprueba tu comprensión

Pregunta documental: ¿Una ronda marcada como completada basta para afirmar que toda la aplicación tuvo éxito?

Respuesta corta: No. Deben examinarse resultado del motor, informe y efectos pertinentes; la guía contempla fallos y resultados parciales, y la revisión estática identifica límites de esa etiqueta. [S16, S17, S19]

Amplía este punto en el curso

Módulo 10 · Seguimiento, medición e incidentes. Ayuda a interpretar qué acredita cada señal y qué falta para cerrar una conclusión.

Módulo 12 · Revisión y mejora. Conecta resultados y limitaciones con las decisiones de la organización.

Fuentes de la lección

S04 · Arquitectura y finalidad

S08 · Datos, estado y espacios de trabajo

S10 · Orquestación de modelos y herramientas

S11 · Revisión de cumplimiento y cálculo de su resultado

S12 · Preguntas frecuentes y límites del recorrido

S13 · Roles, propiedad de sesiones y acceso

S16 · Revisión y acciones de la interfaz

S17 · Aplicación del plan existente

S19 · Trazas y datos de la operación

S21 · Operación y despliegue

El registro completo de archivos aparece al final del documento y en las fuentes del paquete.

Lección 6 · Mapa visual

KUMOSS · INFRAESTRUCTURA CON IA

Sesiones, informes y límites de la observabilidad

Interpreta el progreso, los bloqueos y los resultados sin convertir un estado técnico en una conclusión de negocio.

ANTES DE EMPEZAR

Sesión = trabajo agrupado; ronda = una nueva petición. Observabilidad = información para entender qué ha ocurrido.

01

Aceptación inicial

La API recibe el trabajo de forma asíncrona. La respuesta inicial no demuestra que haya terminado.

02

Estado e informe

Una sesión terminada no demuestra que apply haya tenido éxito. Hay que leer su resultado y sus efectos.

03

Resultado de apply

Éxito, aplicación parcial o fallo requieren comprobación. No hay reintento ni reversión automáticos.

04

Relacionar evidencias

Vincula sesión, código, plan, informe, identidad y trazas. Comprueba después el estado real del servicio.

CLAVE PARA INTERPRETAR

Fechas, firmas o hashes pueden apoyar el registro organizativo. Por sí solos no prueban evidencias completas, inmutables ni suficientes.

Fuente: InditexTech/kumoss · sesiones, resultados y trazas
Rev. 21eb7866ac995f6d6f3159f2ea342e472893638c
José Manuel Hernández Agudo · 30-10-2026 · Lectura estática

Lección 7 · La demostración oficial y su lectura desde el SGIA

Estudia el caso publicado de imágenes e IP fija, su posterior borrado y tres niveles de razonamiento sobre lo que la evidencia permite concluir.

Un caso documentado y atribuible

La guía oficial presenta una sesión sobre una suscripción cloud real: una aplicación necesita almacenar imágenes y utilizar una IP de salida fija. La IP de salida es la dirección que ven los servicios a los que se conecta. Después se solicita borrar la cuenta de almacenamiento. Estudiamos el relato y los momentos publicados por el proyecto; este curso no ha repetido la ejecución. [S03]

Los autores describen una sesión sin guion, con esperas acortadas, datos sensibles ocultos y textos de apoyo añadidos. Indican que código, planes, informes y resultados aplicados no fueron editados. La petición se escribió en español y se muestra en inglés; las tarjetas de notificación son recreaciones de mensajes enviados fuera de pantalla. Estas condiciones forman parte de la interpretación de la demostración.

La procedencia de InditexTech está acreditada. Este ejemplo no documenta una implantación global de Inditex, sus aprobadores efectivos ni un resultado económico de la empresa. Es un caso publicado para comprender el funcionamiento y formular preguntas fundamentadas. [S01, S02]

Seguir la necesidad hasta el resultado

Momento de la guía	Qué describe la fuente	Qué ayuda a comprender
0:10	Petición de imágenes e IP fija y selección de repositorio	La necesidad precede al código.
0:23-0:54	Lectura de contexto, convenciones y ajuste de diferencias de red en el código	La propuesta utiliza información del entorno y reglas.
1:10-1:25	Plan del motor y presentación de informe	Las operaciones y su explicación son objetos revisables.
2:00-2:15	PR, confirmación, aplicación y recursos en consola	Preparar, revisar y ejecutar son hitos separados.
2:46-3:08	Petición de borrar almacenamiento, alto impacto y bloqueo	Los controles pueden detener la aplicación.
3:27-3:44	Revisión y desbloqueo por editor, aplicación y borrado	Hay intervención de roles y un efecto posterior que interpretar.

Los tiempos orientan la lectura de la guía; no son mediciones del tiempo total de ejecución porque se acortaron esperas. El borrado descrito pertenece a la demostración: no es un incidente inventado para la actividad. La dirección de la reconciliación sigue siendo actualizar código a lo observado, incluso cuando aparece dentro del recorrido de Generar. [S03, S06]

Puedes consultar la demostración y sus condiciones para contrastar cada momento.

Qué aporta al sistema de gestión

El SGIA puede aprovechar código, planes, informes, roles, bloqueos y trazas como apoyo a sus procesos. La organización debe determinar responsables, riesgos, criterios y evidencias del uso concreto. Un informe favorable informa una decisión; la autoridad para decidir depende del procedimiento y de las atribuciones correspondientes. [S11, S13, S17]

Referencia de ISO/IEC 42001	Conexión analítica con el caso
5.3 y 7.5	Atribuciones de quienes actúan y documentación de decisiones pertinentes.
8.1; 8.2 y 8.3	Operación, evaluación y tratamiento de riesgos del uso previsto.
9.1 y 9.2	Seguimiento y contraste de evidencias mediante auditoría.
6.3	Planificación cuando cambian el propio SGIA, sus políticas o responsabilidades.
A.6; A.8; A.10	Ciclo de vida; información a interesados; relaciones con terceros, respectivamente.

Estas conexiones orientan preguntas; no son una declaración de conformidad. Un cambio técnico en cloud no constituye automáticamente un cambio del SGIA bajo 6.3. A.8 se refiere a información a partes interesadas; A.10, a terceros. La aplicabilidad de controles requiere el análisis de la organización.

Rúbrica: tres niveles de razonamiento

La pregunta documental de fondo es esta: el controlador exige un plan fijado, lo ejecuta y lo descarta; ¿esa comprobación basta para afirmar autorización organizativa? [S17]

Nivel	Respuesta que ilustra el desempeño
1. Técnico: comprensión inicial	Explica que se ejecuta el plan preparado, pero asume que disponer de él y no ver errores valida todo el proceso.
2. Cumplimiento: criterio fundamentado	Distingue condición técnica y autorización. Identifica criterio aplicable, autoridad y evidencia que tendría que comprobar, sin inventar su ausencia.
3. Auditor sénior / AIGO: criterio de auditoría	Delimita aporte, posible brecha y evidencia disponible; pide información decisiva, propone una medida proporcional si se demuestra la brecha y define cómo verificarla.

AIGO se utiliza aquí como denominación de una función de gobierno de la IA. La rúbrica evalúa respuestas y no concede una acreditación profesional. El tercer nivel no exige automáticamente más

documentos: pide los que permiten resolver la cuestión. Una firma solo se comprueba si la exige el procedimiento aplicable. Si hay autorización bajo política previa, se examinan sus condiciones y la evidencia por cambio; no se atribuye esa automatización de forma nativa a Kumoss.

Comprueba tu comprensión

Pregunta documental: ¿Qué conclusión sobre el bloqueo y el borrado permite la demostración?

Respuesta corta: La guía describe una petición destructiva bloqueada, revisión y desbloqueo por un editor y posterior aplicación que elimina recursos. No permite deducir por sí sola el procedimiento de autorización ni la conformidad de una organización. [S03, S13]

Amplía este punto en el curso

Módulo 5 · Riesgos e impactos. Permite interpretar la utilidad y las consecuencias dentro de un uso y contexto definidos.

Módulo 11 · Auditoría interna. Amplía cómo relacionar una afirmación con la evidencia que realmente la respalda.

Fuentes de la lección

S01 · Organización oficial InditexTech

S02 · Presentación del proyecto

S03 · Demostración publicada y condiciones de la grabación

S06 · Comparación de los modos

S11 · Revisión de cumplimiento y cálculo de su resultado

S13 · Roles, propiedad de sesiones y acceso

S16 · Revisión y acciones de la interfaz

S17 · Aplicación del plan existente

S19 · Trazas y datos de la operación

S20 · Configuración distribuida y valores de clase

S21 · Operación y despliegue

El registro completo de archivos aparece al final del documento y en las fuentes del paquete.

Lección 7 · Mapa visual

KUMOSS · INFRAESTRUCTURA CON IA

Estudio de caso: aportes, brechas y evidencia para el SGIA

Valora la contribución del componente, el riesgo que permanece y la evidencia que necesita una decisión o una auditoría.

ANTES DE EMPEZAR

SGIA = sistema de gestión de la IA. La herramienta aporta mecanismos; la organización conserva autoridad y responsabilidad.

01

La petición publicada

La aplicación pide almacenar imágenes y una IP de salida fija: dirección con la que se presenta ante otros servicios.

02

Un recorrido visible

La demo muestra contexto, código, plan, propuesta en Git, confirmación y aplicación. Esta revisión no repitió la ejecución.

03

Una petición destructiva

Después se pide borrar almacenamiento: la sesión se bloquea; un editor la revisa y retira el bloqueo.

04

Decisión y seguimiento

La autoridad decide antes de aplicar, manualmente o bajo política aplicable; después comprueba el resultado.

CLAVE PARA INTERPRETAR

La demo oficial acorta esperas, oculta datos sensibles y recrea notificaciones. No prueba resultados internos de Inditex ni conformidad del SGIA.

Fuente: InditexTech/kumoss · demostración oficial y controles
Rev. 21eb7866ac995f6d6f3159f2ea342e472893638c
José Manuel Hernández Agudo · 30-10-2026 · Lectura estática

Generar · Recorrido y responsabilidades

KUMOSS · SOLO MODO GENERAR

Generar: de la petición al resultado

IaC = infraestructura como código · apply = ejecutar un plan

1 · Petición + propuesta de IA
Persona: necesidad, repositorio, raíz IaC, proveedor y ámbito.
IA: interpreta el contexto y propone código.

2 · Motor: validar y planificar
OpenTofu o Terraform contrastan el código y anticipan operaciones.

3 · Informes + plan fijado
PR = propuesta de integrar código en Git; es una acción separada.
El plan fijado conserva el artefacto preparado para apply.

BARRERA DE LA ORGANIZACIÓN · AUT = AUTORIDAD DE CAMBIO

Decidir sobre este plan y su contexto
Manual o bajo política previa aplicable y comprobada por cambio.
La API no exige PR previa ni un aprobador distinto.

4 · Acción explícita de apply
El servidor exige plan fijado y sesión desbloqueada.
Ejecuta ese plan sin regenerarlo; el intento consume el plan.

5 · Resultado y seguimiento
Éxito, parcial o fallo: leer el resultado y comprobar el servicio.
No hay reintento ni reversión automática de lo aplicado.

ROLES

developer
Generar y apply en su propia sesión.

panel editor
Puede bloquear o desbloquear la sesión.

El permiso del producto no define por sí solo quién es la AUT.

OTROS MODOS

Drift: termina en Git; no hay apply posterior.

Importar: escribe estado en la ronda; sin apply posterior.

Fuente: InditexTech/kumoss · guías, roles y código del plan fijado
Rev. 21eb7866ac995f6d6f3159f2ea342e472893638c
José Manuel Hernández Agudo · 30-10-2026 · Lectura estática

Fichas en blanco para el alumno

El paquete incluye tres fichas independientes en Word editable y PDF para imprimir. Cada una contiene seis preguntas sin resolver y un dictamen sin seleccionar. Utiliza la rúbrica del monográfico para justificar tus conclusiones.

Actividad 1 · Elegibilidad para automerge en GitOps

Analiza alcance, seguridad, permisos y costes. La delegación se estudia como política organizativa externa; no se atribuye un motor nativo de automerge a Kumoss.

Actividad 2 · Dictamen de reconciliación de drift

Relaciona lo observado con lo autorizado y fundamenta si se acepta su adopción en código o se necesita otro procedimiento.

Actividad 3 · Minimización en trazas

Examina las fuentes de instrumentación y, solo si se aporta, una muestra autorizada y saneada. Distingue riesgo posible de exposición demostrada.

Las fichas están en Actividades/01_Alumno dentro del ZIP. La opción de información insuficiente debe indicar qué falta y cómo podría comprobarse.

Matriz de correspondencia normativa

Correspondencia funcional con ISO/IEC 42001

Relacionar capacidades observadas del producto con preguntas, requisitos y áreas de control que el SGIA debe valorar según su alcance y tratamiento de riesgos.

La selección y aplicabilidad de los controles se justifica en cada SGIA. La cláusula 6.3 se refiere a cambios del propio sistema de gestión. A.8 trata información a interesados y A.10 relaciones con terceros y clientes. La evidencia requerida se determina por el criterio aplicable.

Mecanismo	Cláusulas	Área del Anexo A	Evidencia que se debe valorar
Plan fijado y aplicación	8.1 Operación; 7.5 Información documentada. 6.3 solo si cambia el SGIA.	A.6 Ciclo de vida; A.9 Uso de sistemas de IA, según aplicabilidad.	Relación entre petición, commit, plan identificado, decisión válida, ejecución y estado posterior. Hash o firma cuando el procedimiento los exija. Comprobar cómo se protege la identidad del artefacto.
Revisión de cumplimiento	8.1 Operación; 9.1 Seguimiento y evaluación; 7.5 Información documentada.	A.6 Ciclo de vida; A.9 Uso. A.8 si hay requisitos específicos de información a interesados.	Configuración efectiva, reglas y versiones de prompts activas en Phoenix, entrada evaluada, hallazgos y resultado. Pruebas de eficacia y decisión sobre sus límites. El YAML por sí solo no acredita todas las reglas efectivas.
Identidad y roles OIDC	5.3 Responsabilidades y autoridad; 8.1 Operación; 7.5 Información documentada.	A.3 Organización interna; A.9 Uso. A.10 para responsabilidades de un proveedor de identidad externo, si procede.	Configuración efectiva saneada del emisor y cliente, validación de tokens, asignación y revisión de roles, propiedad y pruebas de permisos. Relación entre permisos técnicos y atribuciones de la organización; nunca adjuntar secretos.
Reconciliación de drift	8.1 Operación; 8.2 y 8.3 Evaluación y tratamiento del riesgo.	A.6 Ciclo de vida; A.9 Uso de sistemas de IA.	Ámbito, código de referencia, diferencias observadas, targets, excepciones y propuesta de diff. Criterio y decisión de aceptar, excluir o corregir. La remediación cloud se evalúa por un procedimiento distinto del modo Drift.
Importación y estado	8.1 Operación; 7.5 Información documentada; 8.2 Evaluación del riesgo.	A.6 Ciclo de vida; A.9 Uso de sistemas de IA.	Alcance autorizado, inventario y exclusiones, correspondencia de recursos e identificadores, cambios de estado y resultados parciales. Protección y recuperación del backend; evidencias del momento en que se autoriza cada efecto.

Mecanismo	Cláusulas	Área del Anexo A	Evidencia que se debe valorar
Trazas y datos del proceso	7.5 Información documentada; 9.1 Seguimiento; 8.2 y 8.3 Riesgo.	A.6 Ciclo de vida; A.9 Uso. A.7 cuando se evalúan datos utilizados por el sistema de IA.	Mapa de datos y destinos, acceso y retención, muestra autorizada y saneada cuando exista, reglas de filtrado y pruebas de minimización. Identificar exactamente qué dato se observó; ausencia de muestra no equivale a fuga demostrada.
Modelos y servicios externos	8.1 Operación; 6.1.2 y 6.1.3 Evaluación y tratamiento del riesgo.	A.10 Relaciones con terceros y clientes.	Versiones y procedencia, dependencias, licencias, condiciones del proveedor, flujos de datos, responsabilidades de actualización y aceptación de riesgos. No atribuir al proyecto la seguridad de toda su cadena sin contraste.
Cambios de políticas y responsabilidades del SGIA	6.3 Planificación de cambios del SGIA; 5.3 Autoridad; 7.5 Información documentada.	A.2 Políticas de IA; A.3 Organización interna, según aplicabilidad.	Cambio del procedimiento o política, motivo, consecuencias, responsables, revisión, versión y comunicación. Si se propone automerge, política previa y comprobación por cambio mediante controles externos; no capacidad nativa acreditada.

Hash, firma y método de autorización se justifican en el procedimiento; esta matriz no los presenta como requisitos universales ni como funciones nativas acreditadas.

IEC Vista previa e índice oficial de ISO IEC 42001 2023

BSI AI Controls Implementation Agenda

Preguntas y respuestas razonadas

Dieciséis preguntas de respuesta única y cuatro ejercicios de razonamiento sobre las fuentes y la demostración publicada. Lee primero la pregunta y contrasta después la respuesta. Este material original del curso no reproduce un banco oficial de certificación.

Pregunta 1 · Necesidad que aborda la demostración oficial

La guía de la demostración oficial comienza con una aplicación que necesita almacenar imágenes y disponer de una IP de salida fija. ¿Qué papel atribuye a Kumoss el recorrido publicado?

A. Entrenar un modelo propio con imágenes de la aplicación antes de crear infraestructura.

B. Sustituir la revisión del cambio por una respuesta de chat que ya constituye su autorización.

C. Traducir la necesidad a una propuesta de infraestructura como código, contrastarla mediante un plan y facilitar su revisión y aplicación explícita.

D. Medir el ahorro y la disponibilidad de todas las instalaciones empresariales del producto.

Respuesta correcta C

La necesidad es de almacenamiento y conectividad: la IP de salida identifica las conexiones de la aplicación ante otros servicios. La guía sigue la lectura del contexto, la preparación de código, un plan del motor y la revisión antes de aplicar. Esto explica qué preparación automatiza Kumoss y para qué se utiliza. La grabación publicada permite estudiar ese recorrido; no aporta un entrenamiento propio ni una evaluación de todas las implantaciones. El vínculo con 8.1 es comprender y controlar la operación; A.9 ayuda a situar las condiciones del uso previsto, según su aplicabilidad.

Por qué se descartan las otras opciones

A. La petición trata de recursos para almacenar imágenes, no de entrenar un modelo con ellas.

B. El recorrido incluye revisión y acciones explícitas; el texto del modelo no acredita autoridad organizativa.

D. Una demostración concreta no mide resultados de todas las instalaciones.

Evidencia y relación con el SGIA

Guía de la demo, necesidad inicial, contexto consultado, código y plan mostrados y pasos de revisión; para evaluar eficacia empresarial harían falta criterios y mediciones adicionales.

Correspondencia funcional: 8.1, A.9. Lección 1.

Sigue la petición de almacenamiento e IP fija hasta el plan, la PR y la aplicación.

see-it-in-action.adoc · What you see, líneas 38-76

Describe la conversión de peticiones en IaC y las integraciones que componen la solución.

about.adoc · Introducción, líneas 4-15

Pregunta 2 · Roles de modelo y componentes de la solución

Al leer el selector de modelos y la arquitectura, ¿qué explicación ayuda a entender cómo se reparte el trabajo de IA sin atribuir al proyecto capacidades no acreditadas?

A. El código asigna ciertos prompts al modelo principal y otros al pequeño; los modelos se configuran mediante LiteLLM y sus roles no prueban entrenamiento propio de Inditex.

B. Cada nombre de agente acredita un modelo distinto entrenado por la organización publicadora.

C. El modelo principal ejecuta directamente OpenTofu sin que intervenga un servicio de aplicación.

D. El modelo pequeño solo comprime trazas y no participa en decisiones del recorrido.

Respuesta correcta A

El selector usa el principal para generación IaC, targets, informes y descomposición de tareas, y el pequeño para los demás tipos de prompt. LiteLLM conecta la aplicación con los modelos configurados. Un rol describe una función en el programa; no identifica quién entrenó el modelo ni prueba concurrencia entre agentes. La operación IaC tiene su servicio y contrato. Para el SGIA, 4.3 ayuda a delimitar el sistema compuesto, y A.10 conecta funcionalmente con las relaciones y responsabilidades de los proveedores elegidos.

Por qué se descartan las otras opciones

B. Confunde un papel de la orquestación con procedencia y entrenamiento de un modelo.

C. La ejecución del motor corresponde al servicio IaC; no se deduce de la selección del LLM.

D. El pequeño también se utiliza en funciones como la revisión de cumplimiento.

Evidencia y relación con el SGIA

Selector por tipo de prompt, configuración efectiva de modelos, proveedor y condiciones del servicio y diagrama de componentes con responsabilidades.

Correspondencia funcional: 4.3, 8.1, A.10. Lección 1.

Selecciona el modelo principal para cuatro tipos de prompt y el pequeño para los demás.

llm_service.py · __select_model, líneas 192-200

Sitúa modelos compatibles con LiteLLM y un ejecutor IaC sustituible por contrato.

about.adoc · Bring your own, líneas 89-100

Pregunta 3 · Del modelo a las herramientas y al resultado técnico

El bucle de un agente termina al obtener su salida estructurada. ¿Qué debe comprobar después el alumno para saber si el código generado superó el criterio técnico de esa etapa?

A. Solo que el resumen del modelo contenga una afirmación de éxito.

B. Que haya terminado la conversación, porque su cierre equivale a una ejecución correcta de la infraestructura.

C. Que el generador haya usado una herramienta de archivos al menos una vez.

D. El resultado del validador que llama el programa; el cierre del bucle y la validación son hitos distintos.

Respuesta correcta D

El orquestador procesa llamadas a herramientas y termina cuando detecta la herramienta de cierre con un resultado. Después, generate_and_validate invoca un validador y acepta el intento cuando result.ok es verdadero. Así se distinguen la propuesta del LLM, las acciones de herramientas y la comprobación técnica posterior. Un cierre o una frase de éxito no contienen por sí solos el resultado del motor. 8.1 se relaciona con el control de estas etapas y 9.1 con interpretar los resultados frente al criterio de comprobación definido.

Por qué se descartan las otras opciones

A. El resumen es contenido del modelo, no el criterio programado del validador.

B. El fin del bucle no demuestra éxito de validación ni de apply.

C. Editar un archivo demuestra una operación, no su corrección técnica.

Evidencia y relación con el SGIA

Salida del bucle, cambios de archivos, entrada y resultado del validador, feedback y etapa alcanzada; no confundirlos con una aplicación cloud.

Correspondencia funcional: 8.1, 9.1, A.6. Lección 2.

Reconoce el cierre por nombre de herramienta y resultado.

llm_service.py · __sentinel_executed, líneas 202-218

Invoca el validador, comprueba result.ok y usa el feedback cuando necesita continuar.

terraform_validation_service.py · generate_and_validate, líneas 146-179

Pregunta 4 · Convenciones y versiones de prompts

Para explicar qué reglas utilizó una ejecución, se dispone de los YAML de semillas del repositorio. ¿Qué información adicional exige la ruta de lectura de prompts examinada?

A. Ninguna: los YAML del repositorio son siempre una copia exacta del contenido efectivo en todos los entornos.

B. Los nombres, el entorno y las versiones de prompts recuperadas de Phoenix en esa ejecución, además de la revisión de las plantillas locales.

C. Únicamente la versión del navegador utilizado por quien inició la sesión.

D. Una prueba de que los modelos fueron reentrenados cada vez que se cambió una convención.

Respuesta correcta B

El fetcher compone un nombre cualificado y pide el prompt a Phoenix por la etiqueta del entorno en tiempo de ejecución. Por eso las semillas versionadas permiten estudiar el diseño, pero no acreditan por sí solas la regla activa de una instalación. Conocer las reglas efectivas ayuda a explicar por qué el agente produjo una propuesta y a reproducir su contexto de revisión. 7.5 conecta con identificar información documentada pertinente; 8.1, con las instrucciones usadas para controlar la operación. Modificar un prompt no demuestra entrenamiento del modelo.

Por qué se descartan las otras opciones

A. Ignora la recuperación por etiqueta y las versiones conservadas en Phoenix.

C. El navegador no identifica las reglas que el núcleo incorpora al prompt.

D. Una actualización de instrucciones no prueba una actualización de pesos del modelo.

Evidencia y relación con el SGIA

Commit de layouts y semillas, nombres cualificados, etiqueta del entorno y versiones efectivamente recuperadas; conservarlas de forma proporcionada al análisis.

Correspondencia funcional: 7.5, 8.1, A.6. Lección 2.

Recupera en ejecución un prompt de Phoenix por nombre y etiqueta de entorno.

_fetcher.py · _PromptFetcher.fetch, líneas 20-36

El entorno selecciona proyecto de trazas y etiqueta de prompts; no activa barreras de seguridad.

config.yaml · environment, líneas 24-30

Pregunta 5 · Validación iterativa y publicación de una rama

En generate_and_validate, ¿qué permite concluir encontrar cambios enviados a la rama de sesión?

A. Que ya se ha integrado la PR en la rama principal y se ha aplicado la infraestructura.

B. Que el intento tuvo que superar el validador antes de publicar cualquier archivo.

C. Que el bucle seguirá sin límite hasta encontrar una propuesta válida.

D. Que se publicaron cambios de ese intento; el commit y push preceden al validador y no acreditan por sí solos su aceptación ni un despliegue.

Respuesta correcta D

La función conserva cambios y hace commit/push antes de llamar al validador. Después devuelve el resultado si es correcto o usa el feedback para otro intento dentro de max_iterations. Si agota el límite, eleva ValidationLoopExceededError. La rama permite revisar el trabajo generado, incluso cuando el intento no supera la comprobación; su existencia no es una aprobación, merge ni apply. La conexión con 8.1 es conocer los puntos de control reales y con 7.5 mantener identificadas las evidencias de cada intento.

Por qué se descartan las otras opciones

A. Publicar en una rama, integrar una PR y aplicar son operaciones distintas.

B. Invierte el orden observado: commit/push aparece antes del validador.

C. El ciclo tiene un límite y una salida de error al agotarlo.

Evidencia y relación con el SGIA

Rama y commit del intento, cambios conservados, resultado y feedback del validador, límite efectivo y estado final del recorrido.

Correspondencia funcional: 7.5, 8.1, A.6. Lección 2.

Itera con límite, publica cambios antes de validar y retorna por result.ok o termina con error.

terraform_validation_service.py · generate_and_validate, líneas 115-183

Pregunta 6 · Plan fijado, revisión y autorización

¿Para qué sirve fijar el plan en Generar y qué límite debe explicar el alumno al relacionarlo con la decisión de aplicar?

A. Firma criptográficamente la decisión de dos aprobadores independientes.

B. Convierte automáticamente en autorizado cualquier plan posterior de la misma sesión.

C. Conserva el artefacto preparado para ejecutarlo sin replanificar; la competencia y la decisión de quien autoriza requieren evidencias adicionales.

D. Permite sustituir el plan por otro equivalente al aplicar sin revisar diferencias.

Respuesta correcta C

La guía explica que apply ejecuta el plan revisado sin regenerarlo. El handler exige un plan fijado y lo descarta al terminar el intento. Esto favorece relacionar preparación y ejecución, pero no identifica por sí solo quién tiene autoridad ni demuestra una aprobación independiente. Tampoco acredita una firma o inmutabilidad criptográfica. Para 8.1 y 7.5 interesa vincular petición, artefacto, decisión y resultado; el método concreto de identificación o firma se determina en el procedimiento aplicable.

Por qué se descartan las otras opciones

A. La fijación examinada no demuestra firma ni doble aprobación.

B. Una decisión referida a un artefacto no cubre automáticamente su sustitución.

D. El recorrido pretende aplicar el artefacto preparado, sin replanificar al ejecutar.

Evidencia y relación con el SGIA

Plan identificado, revisión de código y contexto, decisión de una autoridad competente y registro de la ejecución correspondiente; hash o firma si el procedimiento los requiere.

Correspondencia funcional: 5.3, 7.5, 8.1, A.6. Lección 2.

Comprueba la existencia del plan, llama a apply y descarta el pin en finally.

terraform_apply_handler.py · background_task, líneas 67-89

Describe ejecución sin replanificar, consumo del plan y preparación de otro para la siguiente aplicación.

merge-and-apply.adoc · Apply y Verify, líneas 52-64

Pregunta 7 · Resultado parcial y verificación posterior

La guía distingue Success, Partial y Failed. Sin ejecutar Kumoss, ¿qué debe enseñar el curso sobre un resultado Partial y el siguiente paso operativo?

A. Puede haber recursos ya modificados; la guía no documenta reversión ni reintento automáticos y hace falta comprobar el estado real antes de preparar la actuación siguiente.

B. Partial significa que todos los cambios se han deshecho y el servicio está restaurado.

C. El cierre de la sesión demuestra que la necesidad inicial quedó satisfecha.

D. El siguiente intento reutiliza necesariamente el mismo plan, porque un fallo impide consumirlo.

Respuesta correcta A

El informe puede describir recursos creados, actualizados, destruidos o fallidos. La guía indica que un apply fallido no se reintenta ni revierte automáticamente; el handler descarta el plan incluso en finally. La comprobación posterior debe contrastar qué sucedió y qué funciona, y la actuación siguiente necesita un plan nuevo cuando corresponda. Es una explicación del comportamiento documentado, no un incidente observado. 8.1 y 9.1 apoyan el control y la comprobación; 10.2 se examinaría si un criterio aplicable permite identificar una no conformidad.

Por qué se descartan las otras opciones

B. No hay una restauración automática documentada que sustente esa conclusión.

C. Terminar la sesión y satisfacer la necesidad son criterios distintos.

D. finally consume el pin también si el intento falla.

Evidencia y relación con el SGIA

Resultado detallado del motor y del informe, recursos afectados, estado posterior, comprobación del servicio y decisión sobre la siguiente actuación; no se aportan resultados reales en este ejercicio.

Correspondencia funcional: 8.1, 9.1, 10.2, A.6. Lección 6.

Reconoce resultados parciales, ausencia de reversión/reintento automáticos y consumo del plan.

merge-and-apply.adoc · Apply y Verify, líneas 52-64

Notifica un resultado desfavorable y descarta el plan en finally.

terraform_apply_handler.py · background_task, líneas 82-89

Pregunta 8 · Cuándo utilizar drift y qué efecto produce

Si la necesidad es reconciliar el código con diferencias observadas en recursos ya desplegados, ¿cómo se debe explicar el modo dedicado de drift?

A. Restaura automáticamente la nube a la configuración Git anterior y aplica un plan al terminar.

B. Importa cualquier recurso encontrado al estado, como parte obligatoria de cada reconciliación.

C. Propone cambios en el código para describir lo desplegado; el recorrido dedicado termina sin plan fijado ni apply posterior y no acredita aceptación organizativa de las diferencias.

D. Autoriza el estado observado en cuanto su informe no contenga errores de sintaxis.

Respuesta correcta C

El prompt de descomposición orienta cada operación hacia el código: debe describir la infraestructura observada. La tabla de modos confirma que drift no fija plan ni habilita apply, y tampoco ejecuta la revisión de cumplimiento de Generar. Conservar consistencia técnica no decide si una diferencia es aceptable para el SGIA. Antes de incorporarla, la organización debe valorar origen, efectos y autorización. 8.2 y 8.3 conectan con evaluar y tratar riesgos; 8.1, con controlar la decisión de adopción en código.

Por qué se descartan las otras opciones

A. Invierte la dirección de reconciliación y atribuye apply al modo dedicado.

B. Confunde reconciliar código con la importación que escribe estado.

D. Un resultado técnico no atribuye autoridad para aceptar el estado observado.

Evidencia y relación con el SGIA

Código de referencia, ámbito, diferencias observadas, diff propuesto, excepciones y decisión sobre su adopción; la corrección cloud, si procede, requiere otro recorrido autorizado.

Correspondencia funcional: 8.1, 8.2, 8.3, A.6, A.9. Lección 3.

Orienta las operaciones a modificar código para describir lo desplegado.

task_splitter.jinja · Instrucciones de reconciliación, líneas 8-27

Drift carece de compliance, plan fijado y apply en su recorrido dedicado.

operating-modes.adoc · Comparación, líneas 134-167

Pregunta 9 · Importación: necesidad, alcance y escritura de estado

¿Qué debe conocer el alumno antes de considerar la importación como forma de incorporar recursos existentes a la gestión IaC?

A. Que solo crea una propuesta Git y toda escritura de estado se pospone hasta el merge.

B. Que escribe estado durante la ronda, sin apply posterior en ese modo, y que la versión documenta importación para Azure, GCP y AWS.

C. Que la función importa cualquier proveedor ofrecido en el selector de generación con el mismo alcance.

D. Que, al tratarse de recursos existentes, no tiene efectos relevantes para el control operacional.

Respuesta correcta B

El estado relaciona recursos con su representación IaC y condiciona operaciones posteriores. La tabla declara que la importación lo escribe durante la ronda: una PR revisada después no es un control previo a todas esas escrituras. La nota limita la importación a Azure, GCP y AWS; los otros proveedores indicados terminan sin importar porque no puede listarse el ámbito. El vínculo con 8.1 es identificar y controlar los efectos en el momento adecuado; 7.5 ayuda a conservar el inventario y la evidencia del estado.

Por qué se descartan las otras opciones

A. Aplaza una escritura que la documentación sitúa dentro de la ronda.

C. Extiende soporte de generación a importación y contradice la limitación documentada.

D. Modificar la relación de recursos en state es un efecto que requiere control.

Evidencia y relación con el SGIA

Proveedor y ámbito admitidos, recursos incluidos y excluidos, permisos y decisión previa aplicable, estado anterior y cambios/resultados de importación cuando existan.

Correspondencia funcional: 7.5, 8.1, 8.2, A.6. Lección 3.

Limita la importación a Azure, GCP y AWS.

operating-modes.adoc · Nota de soporte, líneas 13-15

Los modos Import no fijan plan ni habilitan apply porque escriben estado durante la ronda.

operating-modes.adoc · Plan y apply, líneas 155-167

Pregunta 10 · Configuración distribuida y configuración efectiva

Al comparar config.yaml con OrchestrationConfig, ¿qué descripción de la referencia es correcta y útil para comprobar un despliegue?

A. El YAML activa notificaciones, compliance y bloqueo por impacto alto; la clase da false a los dos últimos si se omiten, por lo que hay que identificar los valores efectivamente cargados.

B. Los tres servicios están desactivados en el YAML porque sus clases tienen valores por defecto false.

C. Cambiar environment a production activa automáticamente OIDC y todos los bloqueos.

D. Una captura del YAML de referencia prueba la configuración de cualquier instalación.

Respuesta correcta A

La sección services habilita notifications y deshabilita mapping y authz. Orchestration activa enable_compliance_checker y block_on_high_impact, aunque sus valores de clase son false. El propio YAML dice que environment selecciona etiquetas y proyectos Phoenix sin activar barreras de seguridad. La configuración distribuida y los valores efectivos son objetos distintos: para conocer la operación real hacen falta archivo, parámetros y versión cargados. 7.5, 8.1 y 9.1 conectan con identificar esa configuración y comprobar sus efectos, sin presumir eficacia por un booleano.

Por qué se descartan las otras opciones

B. Sustituye los valores explícitos del YAML por valores de clase.

C. Environment no gobierna barreras de seguridad según los comentarios.

D. Una referencia no identifica lo que cargó una instalación desconocida.

Evidencia y relación con el SGIA

Revisión y ruta del archivo cargado, valores efectivos saneados, parámetros de arranque y comprobaciones autorizadas de los controles pertinentes.

Correspondencia funcional: 7.5, 8.1, 9.1. Lección 4.

Activa notificaciones y las dos comprobaciones; deja mapping y authz desactivados.

config.yaml · Servicios y orchestration, líneas 71-117

Las dos opciones de control tienen valor de clase false.

system_config.py · OrchestrationConfig, líneas 310-321

Pregunta 11 · Qué comprueba el auditor de cumplimiento

¿Qué explicación representa la combinación entre IA y código en la revisión de cumplimiento de Generar?

A. El programa detecta todas las infracciones sin usar juicios de un modelo.

B. El modelo concede la autorización organizativa al devolver passed=true.

C. La revisión se ejecuta obligatoriamente en Generar, drift e importación.

D. Un modelo recibe petición, plan y reglas; el programa calcula passed a partir de las severidades de los hallazgos recibidos, sin garantizar que el modelo haya detectado todas las infracciones.

Respuesta correcta D

ComplianceCheckService retorna un informe vacío si la opción está deshabilitada. Cuando interviene, presenta la primera petición y el plan al modelo con las convenciones. El handler calcula passed comprobando si las violaciones devueltas incluyen severidad error o critical. Ese criterio es programado, pero la lista sobre la que opera proviene de la evaluación del modelo. Por tanto, ni el cálculo acredita exhaustividad ni un resultado favorable constituye la decisión de una autoridad. 8.1 y 9.1 permiten examinar configuración, cobertura, resultados y eficacia del mecanismo.

Por qué se descartan las otras opciones

A. Ignora que los hallazgos son producidos mediante una llamada LLM.

B. Una evaluación técnica no asigna autoridad ni aceptación de riesgo.

C. La comprobación es opcional y la comparación de modos la sitúa en Generar.

Evidencia y relación con el SGIA

Configuración efectiva, petición y plan evaluados, versiones de reglas/prompts, hallazgos y cálculo del resultado; criterios y comprobaciones adicionales para evaluar falsos negativos.

Correspondencia funcional: 8.1, 9.1, A.6, A.9. Lección 4.

Condiciona la revisión a configuración y conserva el informe de la evaluación.

compliance_check_service.py · check, líneas 35-64

Deriva passed de la presencia de severidades error o critical en las violaciones recibidas.

tool_registry_static.py · _handle_compliance_findings, líneas 492-505

Pregunta 12 · Identidad de referencia y alcance de las conclusiones

El config.yaml examinado tiene oidc.issuer_url vacío. ¿Qué conclusión puede formularse sobre la referencia, sin inventar la configuración de una empresa?

A. Que cambiar el entorno a production hará que el emisor se configure por sí solo.

B. Que la referencia deshabilita OIDC y utiliza una identidad local elevada; una instalación concreta debe comprobarse con su configuración efectiva.

C. Que todas las instalaciones de Kumoss carecen de identidad corporativa.

D. Que la identidad local solo puede leer informes y no operar sobre sesiones.

Respuesta correcta B

El archivo advierte que un emisor vacío deshabilita autenticación y hace actuar las peticiones como desarrollador local con roles superiores. La dependencia del servidor deriva a UserService.resolve cuando OIDC no está habilitado. Son hechos de la referencia, no una observación de todos sus usos. Una evaluación concreta necesita comprobar emisor, validación, roles y exposición real. 4.3 ayuda a delimitar el objeto examinado; 5.3 relaciona permisos con responsabilidades, y 8.1 con las condiciones efectivas de operación.

Por qué se descartan las otras opciones

A. El campo environment no activa barreras ni rellena el emisor.

C. Extrapola una configuración distribuida a instalaciones no examinadas.

D. El comentario del YAML describe roles superiores, no solo consulta.

Evidencia y relación con el SGIA

Archivo cargado saneado, estado efectivo de OIDC, mecanismo de validación, identidades y roles asignados y revisión de privilegios; no se aportan datos de una instalación empresarial.

Correspondencia funcional: 4.3, 5.3, 8.1, A.3, A.9. Lección 4.

Distingue environment de barreras de seguridad y deja vacío el emisor OIDC.

config.yaml · environment y oidc, líneas 24-41

Usa resolución local si OIDC no está habilitado; en caso contrario exige y valida bearer.

deps.py · _oidc_enabled y get_current_user, líneas 26-49

Pregunta 13 · Preflight web y autorización en las rutas API

¿Cómo debe enseñarse el control authz por proyecto a partir del wizard, el cliente de autorización y las rutas de apply y merge leídas?

A. Como una comprobación externa que se repite obligatoriamente antes de cada operación API.

B. Como una consulta que siempre incluye el identificador de ámbito cloud y autoriza sus recursos individualmente.

C. Como un preflight opcional del asistente web; si authz está deshabilitado devuelve permiso local, mientras las rutas revisadas aplican otros controles sin repetir esa decisión externa.

D. Como una función nativa de automerge que sustituye las reglas del proveedor Git.

Respuesta correcta C

El wizard llama a authorize antes de continuar. El cliente devuelve authorized=True cuando el servicio está deshabilitado o sin endpoint. La referencia documenta además que la consulta envía cloud, URL de repositorio y ruta IaC, sin scope id. Apply y merge comprueban rol, acceso/propiedad y bloqueo en las rutas examinadas; no repiten allí el control externo por proyecto. Esta delimitación permite valorar dónde debe reforzarse la política de acceso. El vínculo con 8.1 y 5.3 es comprobar los límites del control y la autoridad, sin atribuirle cobertura universal de API.

Por qué se descartan las otras opciones

A. No representa los puntos de llamada observados en las rutas revisadas.

B. El scope id no forma parte del preflight documentado.

D. Authz es un contrato opcional de autorización, no un mecanismo nativo de automerge.

Evidencia y relación con el SGIA

Configuración de authz, contrato e implementación cuando exista, datos enviados por el wizard y controles de cada ruta; distinguir permiso de proyecto, propiedad y decisión de cambio.

Correspondencia funcional: 5.3, 8.1, A.3, A.9. Lección 4.

El asistente consulta authorize y detiene su avance si no obtiene resultado favorable.

initial_information.ts · runInitialInfoWorkflow, líneas 60-83

Retorna permiso local si authz no está configurado y construye el cuerpo cuando sí lo está.

authz_service.py · check, líneas 60-77

Delimita los datos enviados y la ausencia de scope id.

operating-modes.adoc · Authorization preflight, líneas 36-42

Aplica rol, resolución de acceso y bloqueo sin repetir la consulta externa.

terraform.py · apply_infrastructure, líneas 183-204

Comprueba rol, acceso y bloqueo antes de incorporar la PR.

repository.py · complete_pr, líneas 37-56

Pregunta 14 · Bloqueos, panel y propiedad de la sesión

La demo muestra que un usuario de panel con rol editor desbloquea una sesión. ¿Qué alcance puede atribuirse a ese acto al contrastarlo con las dependencias de acceso?

A. El editor pasa automáticamente a ser propietario de la sesión y puede aplicar por su titular.

B. El desbloqueo acredita una evaluación de impacto organizativa y una aceptación formal del riesgo.

C. Un editor debe ser siempre una persona diferente de quien opera, porque el código impone cuatro ojos.

D. El editor tiene capacidad técnica para cambiar el bloqueo; eso no transfiere propiedad ni demuestra por sí solo autoridad organizativa o doble aprobación.

Respuesta correcta D

La ruta administrativa toggle_lock exige panel editor y actualiza el bloqueo. Por separado, assert_session_access permite leer sesiones ajenas con rol de panel y exige propiedad para las operaciones de escritura que usan esa dependencia, como continuar, merge o apply. El desbloqueo administrativo no transfiere esa propiedad. Así se distinguen administrar un control técnico, operar una sesión y disponer de autoridad en el SGIA. La revisión no demuestra una obligación nativa de doble aprobación ni un registro completo de la justificación del desbloqueo. 5.3 y A.3 ayudan a asignar responsabilidades y autoridad; 8.1, a diseñar evidencia y condiciones de uso del desbloqueo.

Por qué se descartan las otras opciones

A. La propiedad se comprueba por separado y no cambia con el desbloqueo.

B. Cambiar un booleano no documenta por sí solo evaluación ni aceptación de riesgo.

C. Los roles no acreditan una segregación obligatoria entre dos personas distintas.

Evidencia y relación con el SGIA

Asignación de roles y titularidad, criterio de bloqueo/desbloqueo, identidad y motivo de la decisión cuando el procedimiento los exija y relación con la operación concreta.

Correspondencia funcional: 5.3, 8.1, A.3, A.9. Lección 4.

Exige panel editor y actualiza el estado de bloqueo.

admin.py · toggle_lock, líneas 96-112

Permite lectura ajena por panel y exige propietario cuando las operaciones que usan esta dependencia solicitan write=True.

deps.py · assert_session_access, líneas 80-90

La guía de la demo muestra revisión, desbloqueo y posterior aplicación por el desarrollador.

see-it-in-action.adoc · Humans approve, líneas 100-107

Pregunta 15 · Datos, trazas y conservación de evidencias

¿Qué enfoque explica dónde se conserva información y cómo examinarla sin dar por probadas la minimización o la inmutabilidad?

A. Tratar Phoenix como único registro, porque contiene también toda la base de datos de dominio y el estado IaC.

B. Distinguir datos de dominio, versiones de prompts y trazas, artefactos y estado; revisar categorías, destinos, acceso y retención efectivos según su finalidad.

C. Considerar que una fecha de creación hace a todos los registros inmutables y no repudiables.

D. Concluir que no se registra contenido sensible porque algunos campos dataclass usan repr=False.

Respuesta correcta B

La documentación separa core-db, phoenix-db, objetos de artefactos, estado y workspace. Phoenix puede recibir mensajes y resultados de herramientas; la exclusión selectiva de campos no constituye un filtro universal. Las trazas ayudan a investigar, pero su presencia no acredita minimización, acceso adecuado ni inmutabilidad. Tampoco una ruta de captura prueba que se haya registrado un secreto concreto. 7.5 y 9.1 conectan con evidencias pertinentes y seguimiento; A.7 se valora cuando se examinan datos utilizados por el sistema de IA, y A.10 cuando intervienen proveedores.

Por qué se descartan las otras opciones

A. La arquitectura conserva dominio, telemetría, artefactos y estado en ubicaciones diferenciadas.

C. Una marca temporal no demuestra protección criptográfica ni ausencia de modificación.

D. La exclusión repr=False es selectiva y no cubre universalmente mensajes y resultados.

Evidencia y relación con el SGIA

Mapa de categorías y destinos, configuración efectiva, reglas de acceso y conservación, mecanismos de protección y, si se aporta, una muestra autorizada saneada con su alcance.

Correspondencia funcional: 7.5, 8.2, 9.1, A.7, A.10. Lección 5.

Separa datos de dominio, telemetría/prompts y objetos de artefactos.

data-and-state.adoc · core-db, phoenix-db y artefactos, líneas 8-52

Excluye selectivamente campos dataclass repr=False.

phoenix_tracer.py · _serialize, líneas 231-255

Construye atributos para mensajes, historial, parámetros y resultados de herramientas.

phoenix_tracer.py · _llm_input_messages_attributes, líneas 403-490

Pregunta 16 · Correspondencia funcional con ISO/IEC 42001

Al relacionar el proyecto con el SGIA, ¿qué mapa funcional mantiene los límites de las cláusulas y áreas del Anexo A indicados en la matriz del curso?

A. 8.1 para controlar la operación; 6.3 cuando cambie el propio SGIA; A.8 para información a partes interesadas y A.10 para relaciones con terceros y clientes, según aplicabilidad.

B. 6.3 para cualquier edición de HCL; A.8 como etiqueta general de cambios y A.10 como sinónimo de control de acceso.

C. Un plan válido demuestra por sí solo conformidad completa con ISO/IEC 42001 y hace innecesaria la evaluación del SGIA.

D. ISO/IEC 42001 impone universalmente un hash, una firma digital y cuatro ojos para cada apply de Kumoss.

Respuesta correcta A

La matriz del curso es una correspondencia funcional, no una certificación. El control de operaciones de infraestructura se conecta principalmente con 8.1. Se añade 6.3 cuando se planifican cambios del sistema de gestión, como políticas, responsabilidades o procesos. A.8 trata información a partes interesadas; A.10, relaciones con terceros y clientes. Su aplicabilidad debe justificarse. Identificar y proteger la relación entre decisión y artefacto es pertinente, pero el método de hash, firma o autorización depende del criterio aplicable; no se deduce un requisito universal ni una función nativa del pin.

Por qué se descartan las otras opciones

B. Confunde cambios técnicos con cambios del SGIA y cambia el significado de A.8 y A.10.

C. Una validación técnica no acredita todos los procesos y requisitos del sistema de gestión.

D. Presenta métodos específicos como obligaciones universales y capacidades acreditadas sin sustento.

Evidencia y relación con el SGIA

Alcance del SGIA, criterio aplicable, evaluación de riesgos, mapa de responsabilidades y justificación de controles; la arquitectura y los artefactos apoyan el análisis técnico.

Correspondencia funcional: 6.3, 7.5, 8.1, A.8, A.10. Lección 7.

Modelos, proveedor de identidad y ejecutor IaC permiten identificar relaciones externas a valorar.

about.adoc · Bring your own, líneas 89-100

Describe servicios de integración y la configuración distribuida que sirve de entrada al análisis funcional.

config.yaml · Contratos y notificaciones, líneas 65-86

Pregunta 17 · Lectura razonada de la demostración oficial

Estudia la guía de la demo oficial, sin instalar ni ejecutar Kumoss. Explica la necesidad de almacenamiento de imágenes e IP de salida fija, el recorrido hasta el plan y la aplicación, y qué cambia cuando se solicita borrar la cuenta de almacenamiento. Separa lo que los autores documentan de lo que esta lectura permite verificar. Indica qué evidencia adicional necesitaría un SGIA para valorar responsabilidades y eficacia, sin inventar una empresa, un incidente ni resultados de ejecución propios.

Respuesta razonada

La necesidad inicial combina almacenamiento para imágenes y conectividad mediante una dirección fija de salida. Según la guía, Kumoss lee contexto, genera código con convenciones, reconcilia diferencias, ejecuta un plan y presenta un informe. El recorrido publicado continúa con revisión de la PR y confirmación de apply. El plan anticipa operaciones; apply es el paso que las ejecuta. La guía sitúa después una petición destructiva, clasificación de impacto alto, bloqueo, notificación, revisión y desbloqueo por un editor, y aplicación por el desarrollador. Esta secuencia permite explicar qué hace cada actor y componente.

Los autores describen una sesión en una suscripción real, con esperas acortadas, datos sensibles ocultos y tarjetas de notificación recreadas; dicen que no editaron código, planes, informes ni resultados aplicados. La respuesta debe atribuir esas condiciones a la publicación. Leer la guía o fotogramas no equivale a repetir la ejecución ni a revisar todos sus artefactos. El bloqueo mostrado tampoco mide todos los falsos negativos posibles, y el término audit trail no demuestra por sí solo inmutabilidad criptográfica.

Para valorar un SGIA pediría alcance y uso previsto, asignación de autoridad, criterio de desbloqueo, configuración efectiva, reglas/prompts y versiones, plan y diff identificados, decisión referida a ese artefacto y comprobaciones posteriores acordes con la necesidad. A.8 se vincularía a información que deban recibir las partes interesadas; la tarjeta recreada no acredita su entrega en otra instalación. 8.1 y 9.1 ayudan a revisar operación y eficacia. No se puede deducir despliegue interno, ahorro, ausencia de incidentes ni conformidad ISO a partir de esta demo.

El caso exige comprender la necesidad, el mecanismo y el momento de cada control. La evidencia principal es la demostración publicada con sus condiciones declaradas, no una ejecución realizada durante el curso ni un expediente organizativo.

Pregunta 17 · Evidencia y relación con el SGIA

Guía y vídeo versionados, condiciones declaradas de grabación y momentos del recorrido. Para ampliar conclusiones: artefactos identificados, configuración, autoridad y mediciones posteriores; no se aportan estos últimos como resultados propios.

Correspondencia funcional: 4.3, 5.3, 8.1, 9.1, A.3, A.6, A.8, A.9. Lección 1.

Declara la suscripción real, edición de esperas, datos ocultos y recreación de notificaciones.

see-it-in-action.adoc · About this recording, líneas 12-24

Documenta la necesidad inicial, preparación, aplicación y recorrido de la petición destructiva.

see-it-in-action.adoc · What you see, líneas 38-107

Permite contrastar la capacidad técnica del rol editor con el acto mostrado en la demo.

admin.py · toggle_lock, líneas 96-112

Claves para el razonamiento

explica almacenamiento/IP fija y distingue contexto, código, plan, revisión y ejecución, con referencias a la guía.

describe el recorrido destructivo y los límites del desbloqueo; atribuye a los autores las condiciones de grabación sin inventar resultados propios.

propone evidencias de autoridad, configuración y eficacia; limita el alcance de las conclusiones y usa A.8 como información a interesados cuando proceda.

Pregunta 18 · A1: límites de la elegibilidad para automerge

La plantilla A1, Análisis de elegibilidad para automerge en GitOps, aporta rutas documentales y una fixture HCL hermética con terraform_data. No incluye un diff cloud pendiente, un plan de ese cambio ni una política de delegación. Explica qué puedes analizar ahora, qué dictamen sobre un cambio concreto queda sin sustento y qué evidencias permitirían reconsiderarlo. Distingue merge, apply y autoridad organizativa; no rellenes una ficha en blanco ni atribuyas automerge nativo a Kumoss.

Respuesta razonada

Con este material se puede analizar el diseño: la fixture declara terraform_data y sus comentarios indican que es una prueba hermética sin descargas de proveedores, red ni credenciales cloud. No es un cambio cloud sometido a decisión, no demuestra que unas etiquetas sean inocuas y su lectura no acredita que una prueba haya sido ejecutada. Las rutas de código permiten describir rol, propiedad y bloqueo para merge, la preparación del plan y el descarte tras apply.

El dictamen sustentable respecto de la elegibilidad de un cambio concreto es información insuficiente. Faltan el objeto de decisión, sus efectos y el criterio de delegación. No debe seleccionarse elegible ni rechazarse un cambio inexistente por una supuesta infracción. Harían falta diff y revisión exactos, plan correspondiente, alcance y efectos funcionales, seguridad, IAM y costes, configuración efectiva y protecciones Git. Para una delegación externa se necesitaría una política competente, vigente y acotada, sus exclusiones y escalado de incertidumbres y una evidencia de su aplicación a ese cambio.

Integrar código en Git y ejecutar un plan son efectos distintos. El permiso técnico para merge no demuestra autorización para apply, y fijar un plan no acredita una firma o la competencia de quien decide. Si se propone automerge, habría que diseñarlo y comprobarlo en una capa externa: la revisión no acredita un mecanismo nativo basado en una política de riesgo. La decisión debe vincularse al artefacto que cubre y revisarse si cambia el diff o el plan. 5.3 y 8.1 orientan responsabilidades y control; 6.3 solo sería pertinente si la introducción de esa política cambia el propio SGIA.

A1 enseña a distinguir análisis documental de una decisión por cambio. Una fixture aporta código real de prueba, pero no sustituye diff, plan, política ni evidencia de elegibilidad. El modelo de respuesta explica los límites sin completar las casillas de la plantilla.

Pregunta 18 · Evidencia y relación con el SGIA

Fuentes A1 y sus límites; para decidir sobre un cambio: diff y plan identificados, configuración y protecciones Git, política externa y decisión de elegibilidad con alcance y responsable.

Correspondencia funcional: 5.3, 6.3, 7.5, 8.1, A.3, A.6, A.9. Lección 2.

Define terraform_data y declara el alcance sin red ni credenciales cloud de la fixture.

main.tf · Fixture hermética, líneas 5-12

Permite leer controles de rol, propiedad y bloqueo para merge.

repository.py · complete_pr, líneas 25-58

Calcula bloqueo y conserva el workspace tras preparar el informe.

terraform_crud_handler.py · Informe y bloqueo, líneas 123-158

Exige el plan, lo aplica y consume su fijación, sin acreditar por sí solo la autorización organizativa.

terraform_apply_handler.py · background_task, líneas 67-89

Claves para el razonamiento

identifica la fixture como material hermético de prueba y limita el análisis a diseño, sin afirmar ejecución ni efectos cloud.

concluye información insuficiente para elegibilidad concreta y pide diff/plan, efectos, política vigente y evidencia por cambio.

separa merge y apply, capacidad técnica y autoridad; trata automerge como propuesta externa y reserva 6.3 para cambios del SGIA.

Pregunta 19 · A2: dictamen documental sobre reconciliación de drift

La plantilla A2, Dictamen de reconciliación de drift, no aporta un informe de ejecución ni un diff observado. Con las fuentes de código y la guía, explica la dirección del cambio, el alcance parcial/completo y las diferencias entre Succeeded, excepciones e incorporación de un estado autorizado. Formula qué puede concluirse ahora y qué haría falta para decidir una adopción en código o proponer una remediación cloud, sin inventar una diferencia de red ni ejecutar operaciones.

Respuesta razonada

Puede concluirse documentalmente que el recorrido dedicado transforma diferencias en cambios de HCL para describir lo observado. Parcial delimita objetivos seleccionados; completo trabaja con la raíz IaC. El modo dedicado no fija un plan para apply ni aplica infraestructura después del merge. La documentación contiene un ejemplo cuya redacción sugiere restaurar reglas, pero el prompt explícito, el tratamiento de diferencias y el flujo leído orientan la reconciliación al código. Por eso debe sostenerse la explicación con las rutas y su alcance, sin convertir esa frase del ejemplo en una restauración cloud acreditada.

La guía permite Succeeded con whitelisted_exceptions: esas diferencias se dejan intencionadamente bajo reglas y no rebajan el resultado si lo demás se reconcilia. En cambio, unreconciled_drift identifica diferencias no resueltas. Ni el resultado favorable ni una excepción técnica demuestran aceptación del riesgo, legitimidad del origen ni autorización para adoptar lo desplegado. Tampoco un resultado de sincronía debe extenderse fuera del ámbito y de las reglas realmente evaluados.

Sin informe, diff y respaldo organizativo, hay información insuficiente para aceptar o rechazar una diferencia concreta. Harían falta código de referencia, fecha y ámbito de observación, plan/diferencias, diff de adopción, targets, excepciones y reglas/versiones, evaluación de efectos y decisión competente. Podría aceptarse todo o una parte delimitada, mantenerse pendiente o rechazarse la adopción con un motivo acreditado. Si se decide corregir la nube, se propondrá otro procedimiento autorizado con responsable, evaluación y comprobación posterior; la ficha no lo ejecuta. 8.1, 8.2 y 8.3 conectan la decisión de código con el control y tratamiento de riesgos, sin declarar una no conformidad por la ausencia de material de ejercicio.

A2 trata una decisión sobre adopción en código y sus límites probatorios. El caso contrasta el sentido técnico de reconciliar con el sentido organizativo de aceptar y hace explícito que no hay una ejecución disponible.

Pregunta 19 · Evidencia y relación con el SGIA

Fuentes A2, ámbito y dirección del diseño. Para dictaminar un caso: observación fechada, código y diff identificados, targets, reglas de excepción, efectos y autorización; no se aporta un caso ejecutado.

Correspondencia funcional: 5.3, 7.5, 8.1, 8.2, 8.3, 9.1, A.6, A.9. Lección 3.

Especifica cambios de código para representar lo desplegado.

task_splitter.jinja · Reconciliación, líneas 8-27

Distingue drift restante, excepciones, Succeeded y ausencia de plan fijado/apply; incluye el ejemplo que debe contrastarse.

remediate-drift.adoc · Report y límites, líneas 59-111

Distingue selección parcial y raíz completa.

operating-modes.adoc · Targets, líneas 120-125

Claves para el razonamiento

explica la dirección hacia HCL y delimita parcial/completo; no atribuye apply ni restauración cloud al modo dedicado.

distingue Succeeded, whitelisted_exceptions y drift restante, sin convertirlos en aceptación organizativa.

limita el dictamen a información insuficiente para una diferencia concreta y enumera evidencias y un procedimiento externo para remediación cloud si procede.

Pregunta 20 · A3: minimización de datos y límites de la evidencia

La plantilla A3, Auditoría de minimización de datos en trazas de Phoenix y OpenTelemetry, se resuelve por defecto con documentación; no incluye trazas reales ni configuración efectiva de una instalación. Explica qué categorías y rutas permite estudiar, qué alcance tiene repr=False y por qué desactivar la consola no prueba que cese OTLP. Propón un dictamen delimitado y una comprobación futura segura, sin buscar secretos, afirmar una fuga ni presentar una medida propuesta como ya implantada.

Respuesta razonada

El código permite estudiar metadatos de sesión y usuario, cloud, repositorio, raíz IaC y rama, además de mensajes, historial, nombres y parámetros de herramientas y sus resultados o errores. Esas categorías ayudan a investigar el proceso, pero pueden transportar contenido operativo y deben valorarse según finalidad y necesidad. Una ruta de serialización muestra capacidad de captura; no acredita que una credencial o dato personal concreto se haya registrado.

_serialize omite campos dataclass declarados repr=False, pero deja pasar strings y serializa otros objetos por rutas distintas. Además, el constructor de atributos de mensajes incorpora texto y resultados de herramientas. La exclusión es selectiva, no una garantía general de eliminación de secretos. En el inicializador, el interruptor de consola condiciona ConsoleSpanExporter, mientras que el procesador OTLP se añade aparte. Por tanto, consola desactivada no demuestra exportación OTLP desactivada. La guía recomienda controles de acceso y conservación para Phoenix; eso no prueba que estén implantados en un despliegue no examinado.

El dictamen debe limitarse al diseño: se han identificado categorías y aspectos que requieren medidas o evidencias, pero hay información insuficiente para juzgar la minimización efectiva de una instalación. Propondría justificar campos por finalidad, excluir o transformar contenido innecesario, restringir destinatarios y accesos y definir retención. Cada propuesta indicará responsable y criterio de eficacia. La comprobación futura podría usar una revisión de configuración o una prueba autorizada con datos sintéticos y una muestra saneada; no hay que obtener secretos reales para resolver la ficha. 7.5 y 9.1 orientan evidencias y evaluación; A.7 se valora para los datos del sistema de IA y A.10 para responsabilidades de terceros cuando corresponda. Un indicio futuro se remitiría por su referencia y cauce organizativo, sin declararlo ahora incidente confirmado.

A3 enseña a separar capacidad de captura, captura observada y minimización efectiva. El código y la guía sustentan un análisis de diseño y propuestas comprobables, pero no un hallazgo sobre datos de una instalación no examinada.

Pregunta 20 · Evidencia y relación con el SGIA

Fuentes A3, mapa de categorías y destinos posibles, limitaciones de serialización y recomendaciones. Para ampliar: configuración efectiva saneada, políticas y muestra autorizada saneada; no se solicitan secretos ni nuevas capturas.

Correspondencia funcional: 7.5, 8.1, 8.2, 8.3, 9.1, A.7, A.9, A.10. Lección 5.

Construye metadatos de sesión, usuario y contexto.

phoenix_tracer.py · __metadata_attributes, líneas 68-91

Excluye campos dataclass repr=False de forma selectiva.

phoenix_tracer.py · _serialize, líneas 231-255

Añade texto, historial y llamadas/resultados a los atributos de mensajes.

phoenix_tracer.py · _llm_input_messages_attributes, líneas 403-490

Separa la consola opcional del procesador OTLP añadido al proveedor.

_initializer.py · __init_providers, líneas 61-86

Explica contenidos y controles recomendados de acceso y destino, sin acreditar una instalación concreta.

operations.adoc · Phoenix, líneas 59-69

Claves para el razonamiento

identifica categorías y rutas y distingue capacidad documental de captura efectiva; no inventa exposición ni resultados de trazas.

delimita repr=False y diferencia exportador de consola de OTLP con referencias de código.

emite dictamen acotado, propone medidas proporcionales con responsable y verificación futura segura usando configuración o datos sintéticos autorizados.

Cierre del monográfico

Qué has aprendido a distinguir

Kumoss convierte una necesidad expresada en lenguaje natural en trabajo sobre infraestructura como código. Coordina modelos y herramientas, utiliza el contexto disponible y conserva resultados para revisar. El aporte observable es ese recorrido técnico; el ahorro de tiempo, la reducción de incidentes y la calidad en producción requieren mediciones que esta revisión no contiene.

Generar prepara código y un plan que puede quedar fijado para aplicar. Drift ajusta código a la infraestructura observada y termina en el repositorio, sin cloud apply. Importar incorpora recursos existentes y escribe estado durante la ronda, antes de la PR posterior. Estas diferencias indican qué efectos hay que conocer y cuándo corresponde revisarlos.

La propuesta del modelo, el plan del motor, el resultado del programa y la decisión de una persona tienen funciones distintas. Un plan válido no demuestra ejecución; una incorporación en Git no demuestra aplicación; una etiqueta de completado no sustituye el resultado del motor; un informe favorable no establece por sí solo autorización organizativa. La configuración efectiva, los accesos y la conservación de información condicionan todo el recorrido.

Aplicar el razonamiento del SGIA

Para estudiar un uso concreto, empieza por su finalidad, versión y alcance. Identifica qué modo se utiliza y qué cambia durante la ronda. Relaciona las capacidades técnicas con las responsabilidades y el procedimiento aplicable. Después pregunta qué evidencia permite afirmar que el control funcionó y qué información falta para decidir.

Pregunta de cierre	Evidencia que permitiría profundizar en una implantación
¿Sobre qué se actuó?	Repositorio, ruta IaC, ámbito, proveedor y versiones pertinentes.
¿Qué se preparó y qué se ejecutó?	Código, plan correspondiente, operación y resultado del motor.
¿Qué control estaba activo?	Configuración cargada, reglas efectivas, roles y bloqueos.
¿Quién podía decidir?	Atribuciones y criterio de autorización aplicables, con su registro pertinente.
¿Qué ocurrió después?	Estado resultante, diferencias pendientes y decisión de seguimiento.
¿Dónde quedó la información?	Destinos de datos, accesos y condiciones de conservación y recuperación.

Las evidencias enumeradas son preguntas de evaluación, no documentos que esta revisión atribuya a una empresa. Una ausencia de información en el repositorio del producto no prueba una no conformidad organizativa. Si el procedimiento requiere una firma o un mecanismo específico, se comprobará esa condición; no se prescribe como requisito universal.

La demostración oficial permite practicar con una necesidad y un borrado publicados, sin inventar incidentes o resultados. La rúbrica pide avanzar de describir operaciones a justificar criterios y conclusiones con evidencia proporcionada. Las fichas de actividad y la matriz de correspondencia del paquete continúan ese trabajo.

Alcance que debes conservar al comunicar el estudio

Las conclusiones corresponden a una revisión estática dirigida del commit 21eb7866ac995f6d6f3159f2ea342e472893638c. No se han ejecutado servicios, modelos, pruebas ni operaciones cloud. El caso audiovisual se atribuye a la documentación oficial. El análisis aporta comprensión del diseño y de sus límites; para valorar una implantación o la conformidad de su SGIA se requieren sus evidencias y criterios propios.

Fuentes y versión del estudio

Repositorio: InditexTech/kumoss.

Revisión analizada: 21eb7866ac995f6d6f3159f2ea342e472893638c. Fecha de preparación del material: 30/10/2026.

El análisis es una lectura estática dirigida. Las fuentes del proyecto se conservan en el ZIP con sus avisos y licencia. No se han ejecutado Kumoss, modelos o servicios cloud para esta entrega.

S01 · Organización oficial InditexTech

S02 · Presentación del proyecto · README.md

S03 · Demostración publicada y condiciones de la grabación · see-it-in-action.adoc

S04 · Arquitectura y finalidad · about.adoc

S04 · Arquitectura y finalidad · architecture.adoc

S05 · Cómo se formula y tramita una petición · make-a-request.adoc

S06 · Comparación de los modos · operating-modes.adoc

S07 · Vocabulario oficial · glossary.adoc

S07 · Vocabulario oficial · request.adoc

S07 · Vocabulario oficial · pinned-plan.adoc

S08 · Datos, estado y espacios de trabajo · data-and-state.adoc

S09 · Papeles de modelo y requisitos · llm-providers.adoc

S10 · Orquestación de modelos y herramientas · llm_service.py

S10 · Orquestación de modelos y herramientas · tool_service.py

S11 · Revisión de cumplimiento y cálculo de su resultado · compliance_check_service.py

S11 · Revisión de cumplimiento y cálculo de su resultado · tool_registry_static.py

S12 · Preguntas frecuentes y límites del recorrido · faq.adoc

S13 · Roles, propiedad de sesiones y acceso · roles-and-permissions.adoc

S13 · Roles, propiedad de sesiones y acceso · deps.py

S13 · Roles, propiedad de sesiones y acceso · terraform.py

S13 · Roles, propiedad de sesiones y acceso · repository.py

S13 · Roles, propiedad de sesiones y acceso · auth.py

S13 · Roles, propiedad de sesiones y acceso · authz_service.py

S13 · Roles, propiedad de sesiones y acceso · initial_information.ts

S14 · Composición de instrucciones · template_service.py

S14 · Composición de instrucciones · _fetcher.py

S15 · Generación, validación y fijación · terraform_validation_service.py

S15 · Generación, validación y fijación · terraform_crud_handler.py

S16 · Revisión y acciones de la interfaz · merge-and-apply.adoc

S16 · Revisión y acciones de la interfaz · PrApprovalView.tsx

S17 · Aplicación del plan existente · terraform_apply_handler.py

S17 · Aplicación del plan existente · engine.py

S18 · Recorrido de importación y escritura de estado · terraform_import_handler.py

S18 · Recorrido de importación y escritura de estado · terraform_import_service.py

S19 · Trazas y datos de la operación · phoenix_tracer.py

S19 · Trazas y datos de la operación · operations.adoc

S20 · Configuración distribuida y valores de clase · config.yaml

S20 · Configuración distribuida y valores de clase · system_config.py

S21 · Operación y despliegue · hardening.adoc

S21 · Operación y despliegue · deploy-to-production.adoc

S22 · Licencia, avisos y motor opcional · LICENSE

S22 · Licencia, avisos y motor opcional · NOTICE

S22 · Licencia, avisos y motor opcional · Dockerfile

S22 · Licencia, avisos y motor opcional · about.adoc

S23 · Historial de cambios y versión · CHANGELOG.md

S23 · Historial de cambios y versión · pyproject.toml

S23 · Historial de cambios y versión · roadmap.adoc