



Manual de UltraGX

Instalación, licencia y uso de UltraGX: de la descarga a una KB documentada y lista para migrar.

Versión 0.2.8 · ultra-gx.com

Contenido

01 Introducción

02 Instalación

03 Primer inicio y licencia

04 La aplicación

05 Documentar una KB

06 Leer el resultado

07 Narrativa con IA

08 Notas del usuario

09 Migrar con un agente de IA

10 Línea de comandos

11 Solución de problemas y preguntas frecuentes

12 Glosario

Introducción

UltraGX lee una Knowledge Base (KB) de GeneXus, de la versión 7.5 a la 18, y deja todo lo que sabe de ella en archivos abiertos: documentos Markdown para leer, JSON para procesar y un sitio web navegable que funciona sin conexión. El resultado sirve para tres cosas:

- **Entender el sistema** sin abrir el IDE de GeneXus: qué hace cada objeto, qué reglas de negocio aplica, qué tablas lee y escribe, quién lo llama y a quién llama.
- **Conservar el conocimiento** en un formato que se puede versionar, buscar y compartir, aunque la gente que construyó el sistema ya no esté.
- **Migrar**: la salida está pensada para que un agente de IA (Claude Code, Codex, Cursor, Antigravity) regenere la aplicación en otra tecnología, módulo por módulo, sin volver a interpretar GeneXus desde cero.

Qué hace, paso a paso

1. **Lee la KB** desde un export XPZ/XML, desde un export de texto o, en Windows con GeneXus instalado, abriendo la KB directamente con el conector.
2. **Analiza cada objeto** con reglas deterministas: tipo, estructura, reglas, eventos, variables, navegación, accesos a datos y referencias. No hay adivinanzas de un modelo de lenguaje en esta etapa: cada conclusión queda atada a la evidencia que la produjo.
3. **Relaciona los objetos**: arma el grafo de dependencias, el modelo de datos, los árboles de llamadas, los diagramas de flujo de datos y los *clusters* (grupos funcionales de objetos que trabajan juntos).
4. **Escribe la documentación**: un documento por objeto, el modelo de datos, los clusters, una hoja de ruta de reconstrucción y el visor web.
5. **Opcionalmente, agrega prosa con IA** (la *narrativa*): un modelo de lenguaje redacta explicaciones y el motor verifica que cada nombre citado exista en el objeto antes de publicarla.

Solo lectura

UltraGX **nunca modifica la KB**. El conector abre la Knowledge Base en modo lectura y no tiene ningún camino de escritura; con un XPZ o un export de texto ni siquiera hace falta tener GeneXus instalado. Todo lo que genera se escribe en una carpeta de salida aparte.

Ediciones

	Demo	Completa
Duración	10 días desde la instalación	según la licencia
Inventario, grafo, modelo de datos, clusters, árboles de llamadas y diagramas	toda la KB	toda la KB

	Demo	Completa
Documento detallado por objeto (reglas, pseudocódigo, SQL implícito, interfaz, código original)	25 objetos elegidos por el motor	todos
Knowledge packs, planes de reconstrucción por cluster, narrativa con IA	solo para la muestra	todos
Conexión a internet	necesaria en cada ejecución (verificación de la licencia)	no hace falta

En la demo, el resto de los objetos recibe un documento resumido con su identidad, ubicación, perfil, confianza, cluster y referencias, y un aviso de que el detalle está en la edición completa.

Qué datos salen de tu máquina

- **El análisis de la KB se hace entero en tu equipo.** El código, las reglas y la estructura de tus objetos no se envían a ningún servidor de UltraGX.
- **Licencia demo:** al registrarte mandás tus datos de contacto; en cada ejecución el servicio de licencias recibe el identificador de la licencia, el id de la máquina, **cuántos** objetos tiene la KB (solo cantidades) y los **nombres** de los objetos propuestos para la muestra.
- **Licencia completa:** no hay comunicación con el servicio después de activarla.
- **Narrativa con IA:** si la usás, el documento de cada objeto se envía al proveedor que elijas (Anthropic, OpenAI, Google, etc.) con tu propia API key. Si preferís que nada salga de la red, usá un modelo local (Ollama o LM Studio). Ver [Narrativa con IA](#).

Cómo leer este manual

- **Si estás probando la demo:** [Instalación](#) → [Primer inicio y licencia](#) → [Documentar una KB](#) → [Leer el resultado](#).
- **Si ya tenés licencia:** [Documentar una KB](#) → [Narrativa con IA](#) → [Notas del usuario](#).
- **Si sos parte del equipo de migración:** [Leer el resultado](#) → [Migrar con un agente de IA](#) → [Línea de comandos](#).

Instalación

Todas las descargas están en ultra-gx.com. Al lado de cada archivo se publica su huella SHA-256, por si querés verificar que la descarga llegó entera.

Requisitos

Sistema	Versión mínima	Qué más necesita
Windows	Windows 10 versión 1809 o Windows 11, 64 bits	El runtime Microsoft Edge WebView2 (el instalador lo agrega si falta)
Linux	Debian 12, Ubuntu 22.04, Fedora 39 o equivalentes, 64 bits	libwebkit2gtk-4.1 (o 4.0) y GTK 3
macOS	macOS 13, Apple Silicon o Intel	Nada

No hace falta instalar Python, Node ni .NET: el motor y la aplicación vienen compilados.

Para leer una KB en vivo (sin exportarla antes) hace falta Windows con GeneXus X Evolution 1 a 18 instalado en la misma máquina. Para cualquier otro caso alcanza con un export XPZ/XML o un export de texto, y ahí da igual el sistema operativo.

Espacio en disco: una KB de 700 objetos genera unos 100 MB de documentación; una de 5.000 objetos, alrededor de 1 GB.

Windows

Con el instalador (recomendado)

1. Descargá `UltraGX-<versión>-win-x64-setup.exe`.
2. Ejecutalo. Si Windows muestra **"Windows protegió su PC"** (SmartScreen), hacé clic en **Más información** y después en **Ejecutar de todas formas**. El aviso aparece porque el instalador todavía es nuevo para SmartScreen, no porque haya un problema con el archivo.
3. **Idioma:** elegí English, Español o Português. Ese idioma se usa para el asistente, para el contrato de licencia que se muestra y como idioma inicial de la aplicación y de los documentos que genera. Los dos se pueden cambiar después.
4. **Contrato de licencia:** leelo y aceptalo para continuar.
5. **Carpeta de destino:** por defecto `C:\Users\<usuario>\AppData\Local\Programs\UltraGX`. La instalación es por usuario y no pide permisos de administrador.
6. **Tareas adicionales** (las dos opcionales):
 - *Crear un acceso directo en el escritorio.*
 - *Abrir exports de GeneXus (.xpz) con UltraGX:* con esta opción, al hacer doble clic en un `.xpz` se abre UltraGX con la pestaña **Ejecutar pipeline** lista para documentarlo.
7. **Instalar.** Si al equipo le falta WebView2, el instalador lo descarga e instala en este paso (hace falta conexión a internet).
8. Al terminar, dejá marcada **Ejecutar UltraGX** para abrirlo y seguí con [Primer inicio y licencia](#).

En Windows el instalador incluye también el **conector de GeneXus**, que permite leer una KB en vivo. Si en la máquina no hay GeneXus instalado, el conector queda inactivo y no molesta.

Linux

Elegí el formato que corresponda a tu distribución:

```
# Debian / Ubuntu
sudo apt install ./ultragx-<versión>-amd64.deb

# Fedora / RHEL
sudo dnf install ./ultragx-<versión>-1.x86_64.rpm
```

Los paquetes instalan la aplicación en `/opt/ultragx`, el comando `ultragx` y la entrada en el menú de aplicaciones, y declaran las dependencias de WebKitGTK, así que el gestor de paquetes las instala solo.

AppImage (sin instalar): dale permiso de ejecución y ábrila. En este caso las dependencias tienen que estar presentes en el sistema:

```
sudo apt install libwebkit2gtk-4.1-0      # Debian / Ubuntu
chmod +x UltraGX-<versión>-x86_64.AppImage
./UltraGX-<versión>-x86_64.AppImage
```

En Linux la fuente "KB en vivo" no está disponible (el conector necesita GeneXus, que es de Windows): exportá la KB a XPZ o a texto en la máquina donde está GeneXus y documentala acá.

macOS

1. Descargá el `.dmg` que corresponda a tu Mac: `osx-arm64` para Apple Silicon (M1 en adelante), `osx-x64` para Intel.
2. Abrilo y arrastrá **UltraGX** a **Aplicaciones**.
3. La versión de macOS no está firmada con un Developer ID de Apple, por eso macOS la bloquea la primera vez. Hacé doble clic en **Allow UltraGX.command** de la imagen de disco (o ejecutá `xattr -cr /Applications/UltraGX.app` en Terminal) y abrí UltraGX. Si macOS insiste, andá a **Configuración del Sistema → Privacidad y seguridad** y hacé clic en **Abrir igualmente** junto al mensaje de UltraGX.

Igual que en Linux, en macOS se trabaja con exports XPZ o de texto.

Actualizar

Descargá la versión nueva e instalala encima de la anterior (en Windows, el instalador reemplaza los archivos en la misma carpeta). La configuración, las conexiones de narrativa, la licencia y las salidas generadas se conservan.

Las salidas generadas con una versión anterior se pueden seguir abriendo. Para aprovechar las mejoras del motor nuevo, volvé a ejecutar el pipeline sobre la misma KB: las notas del usuario y la

narrativa ya generada se mantienen.

Desinstalar

- **Windows:** *Configuración* → *Aplicaciones* → *Aplicaciones instaladas* → *UltraGX* → *Desinstalar*.
- **Linux:** `sudo apt remove ultragx` o `sudo dnf remove ultragx` ; la Applmage se borra y listo.
- **macOS:** arrastrá UltraGX desde Aplicaciones a la papelera.

Desinstalar **no borra** tus datos:

Qué	Dónde
Configuración, conexiones de narrativa (incluidas las API keys) y licencia	%APPDATA%\UltraGX en Windows, ~/.config/UltraGX en Linux y macOS
Salidas generadas	la carpeta de salida que elegiste
Registro del período de demo	se conserva a propósito: reinstalar no reinicia los 10 días

Si querés quitar todo, borrá también la carpeta de configuración y tu carpeta de salida. Recordá que la configuración guarda tus API keys.

Primer inicio y licencia

UltraGX necesita una licencia para ejecutar el motor. La primera vez que abrís la aplicación sin licencia aparece el formulario **Empezá tu demo de 10 días**. Desde ahí podés pedir la demo o, si ya compraste la edición completa, usar tu código.

Empezá tu demo de 10 días

UltraGX documenta toda tu Knowledge Base y 25 objetos en detalle. Contanos quién sos y te mandamos un código de verificación por correo.

Nombre	Apellido
Ana	Pérez
Empresa	Correo de la empresa
Mi Empresa S.A.	ana.perez@miempresa.com
Rol	País
Gerente de sistemas	Uruguay
Teléfono	Versión de GeneXus
	18
Objetos en la Knowledge Base (aproximado)	
3200	

[Mandame el código](#) [Ya tengo un código de licencia](#)

Usamos tus datos para contactarte por la demo y la migración. Nada de tu Knowledge Base sale de tu máquina: el servicio solo ve la licencia, el id de máquina y los nombres de los objetos propuestos para la muestra.

Formulario de registro de la demo

Pedir la demo

1. Completá el formulario. Son obligatorios **Nombre, Apellido, Empresa y Correo de la empresa**; el resto (**Rol, País, Teléfono, Versión de GeneXus, Objetos en la Knowledge Base**) nos ayuda a preparar la conversación sobre la migración.
2. Usá el **correo de tu empresa**. La demo se entrega a empresas que están evaluando una migración y el servicio rechaza los correos personales (Gmail, Outlook, Hotmail, Yahoo, iCloud y similares).
3. Hacé clic en **Mandame el código**. En un par de minutos llega un correo con un código de seis dígitos.
4. Escribí el código y hacé clic en **Activar la demo**. El código vale por **15 minutos**.

Empezá tu demo de 10 días

UltraGX documenta toda tu Knowledge Base y 25 objetos en detalle. Contanos quién sos y te mandamos un código de verificación por correo.

Mandamos un código de seis dígitos a ana.perez@miempresa.com. Vale por 15 minutos.

4 8 2 9 1 3

Activar la demo

[Volver](#)

¿No llegó? Mirá la carpeta de correo no deseado, o volvé y probá con otra dirección.

Usamos tus datos para contactarte por la demo y la migración. Nada de tu Knowledge Base sale de tu máquina: el servicio solo ve la licencia, el id de máquina y los nombres de los objetos propuestos para la muestra.

Paso del código de verificación

Si el correo no llega, mirá la carpeta de correo no deseado. Si sigue sin aparecer, hacé clic en **Volver** y pedí el código otra vez (o probá con otra dirección de la empresa).

Cuando la demo queda activa, el formulario se cierra y la aplicación muestra el aviso **Edición DEMO** en la barra superior.

Qué incluye la demo

- **10 días** contados desde la instalación en esa máquina. Reinstalar o pedir otro código no reinicia el plazo.
- Una demo por máquina y una por correo.
- La KB completa se analiza y se documenta a nivel de inventario, modelo de datos, grafo, clusters, árboles de llamadas y diagramas. **25 objetos** reciben el documento detallado; los elige el motor entre los objetos de negocio más conectados, repartidos por tipo, para que la muestra sea representativa. Los mismos 25 se mantienen aunque ejecutes la KB varias veces.
- Cada ejecución **necesita internet** para consultar el servicio de licencias. Si no hay conexión, la ejecución termina igual pero sin objetos detallados, y el aviso queda en el README y en el visor.

Activar la edición completa

Cuando compres la licencia vas a recibir un código con la forma `ULGX-XXXX...`.

- **En el primer inicio:** en el formulario hacé clic en **Ya tengo un código de licencia**, pegá el código y hacé clic en **Activar con un código**.
- **Con la aplicación ya en uso** (por ejemplo, al pasar de la demo a la completa): andá a **Configuración → Licencia**, pegá el código en **Código de activación** y hacé clic en **Activar con un código**. El campo **URL del servicio de licencias** se deja vacío salvo que te indiquemos otra cosa.

La activación ata la licencia a esa máquina. Después de activar, la edición completa no vuelve a conectarse al servicio.

Instalar un archivo de licencia

En instalaciones sin acceso a internet te podemos entregar un archivo `ultragx.lic` en lugar de un código:

1. En **Configuración → Licencia** copió el **Id de esta máquina** y mandáenoslo.
2. Cuando recibas el archivo, hacé clic en **Instalar archivo de licencia...** y elegilo.

El archivo queda en `%APPDATA%\UltraGX\ultragx.lic` (Windows) o `~/.config/UltraGX/ultragx.lic` (Linux y macOS).

Licencia

Estado	válida
Edición	completa
Titular	Mi Empresa S.A. <sistemas@miempresa.com>
Vence	2036-09-04
Archivo de licencia	%APPDATA%\UltraGX\ultragx.lic
Id de esta máquina	3f9c2a71b8e4d05c (envialo al propietario para obtener una licencia atada a esta máquina)

Activar con un código

Configuración, bloque Licencia

Ver el estado de la licencia

Configuración → Licencia muestra:

Campo	Qué significa
Estado	válida, o el motivo por el que no lo es
Edición	DEMO con la cantidad de objetos detallados y el servicio que consulta, o completa
Titular	empresa y correo a los que se emitió
Vence	fecha de vencimiento, o nunca
Archivo de licencia	dónde está instalada
Id de esta máquina	identificador que se usa para atar la licencia

Mensajes del registro y la activación

Mensaje	Causa	Qué hacer
Usó el correo de tu empresa...	Pusiste un correo personal	Registrate con la dirección de la empresa
Este correo ya activó una demo...	Ese correo ya se usó	Escribinos a contact@ultra-gx.com si necesitás más tiempo
El período de demo en esta máquina terminó...	Pasaron los 10 días	Escribinos para seguir con la edición completa

Mensaje	Causa	Qué hacer
Código incorrecto...	El código no coincide	Revisá el último correo recibido y escribilo de nuevo
El código venció...	Pasaron más de 15 minutos	Volver y pedir uno nuevo
Demasiados códigos incorrectos...	Cinco intentos fallidos	Volver y empezar de nuevo
Demasiados intentos desde esta red...	Muchas solicitudes seguidas desde la misma red	Esperá un rato y probá de nuevo
Revisá los campos...	Falta un campo obligatorio o no es válido	Completá nombre, apellido, empresa y un correo válido
No se pudo contactar al servicio de licencias...	Sin internet, o un proxy/firewall bloquea <code>licenses.ultra-gx.com</code>	Verificá la conexión; en redes corporativas pedí que habiliten HTTPS a <code>licenses.ultra-gx.com</code>

La aplicación

La ventana de UltraGX tiene una barra superior con seis pestañas y, a la derecha, la **carpeta de salida** activa con el botón **Carpeta de salida...** para cambiarla. Cada ejecución del pipeline crea una subcarpeta dentro de esa carpeta; conviene elegir una en un disco con espacio y fuera de carpetas sincronizadas (OneDrive, Dropbox), porque una KB grande genera miles de archivos.

La primera vez, si no hay carpeta de salida definida, la aplicación abre directamente **Configuración**.

Proyectos

UltraGX

Proyectos

Visor

Ejecutar pipeline

Narrativa

Tareas

Configuración

h:\models\UltraGX\dist\manual-shots

Carpeta de salida...

Proyectos relevados

Actualizar

Abrir carpeta

Cada carpeta bajo la raíz de salida que contiene una ejecución del pipeline. Elegí una para abrir su visor.

Proyecto	Fuente	Entrada	Objetos	Clusters	Conf. media	Aceptación	Narrativa ok	Modificado	
FacturasDemo	xpz	h:\models\UltraGX\dist\manual-input\FacturasDemo.xml	696	324	82.0	passed		2026-09-16 18:02	Abrir Regenerar visor Carpeta

Pestaña Proyectos

Lista cada carpeta bajo la carpeta de salida que contiene una ejecución del pipeline:

Columna	Qué muestra
Proyecto	nombre de la subcarpeta
Fuente / Entrada	de dónde se leyó la KB (<code>xpz</code> , <code>gxtext</code> , <code>live</code>) y la ruta
Objetos / Clusters	cantidad de objetos documentados y de grupos funcionales
Conf. media	confianza media del análisis, de 0 a 100
Aceptación	resultado del control de calidad final (<code>passed</code> , <code>warning</code> , <code>failed</code>)
Narrativa ok	objetos con prosa de IA publicada
Modificado	fecha de la última ejecución

Abrir lleva el proyecto al **Visor**; **Regenerar visor** reconstruye su sitio web; **Carpeta** lo abre en el explorador de archivos. **Actualizar** vuelve a leer la carpeta de salida y **Abrir carpeta** abre la carpeta de salida entera.

Visor

Muestra el sitio web generado para el proyecto elegido en la lista desplegable. Está explicado en detalle en [Leer el resultado](#).

- **Recargar**: vuelve a cargar la página (útil mientras corre la narrativa, que va actualizando los documentos).
- **Regenerar visor**: reconstruye el sitio desde los archivos de la salida, sin volver a analizar la KB.
- **Abrir en el navegador**: abre el mismo visor en tu navegador. Desde el navegador se puede leer todo, pero las notas del usuario solo se agregan desde la aplicación.

Ejecutar pipeline

Es donde se documenta una KB: se elige la fuente, la entrada, el nombre del proyecto y el idioma de los documentos. Ver [Documentar una KB](#).

Narrativa

Agrega prosa escrita por un modelo de lenguaje a los documentos de un proyecto ya generado, con conexiones a distintos proveedores. Ver [Narrativa con IA](#).

Tareas

UltraGX

Proyectos

Visor

Ejecutar pipeline

Narrativa

Tareas

Configuración

I:\modelos\UltraGX\dist\manual-shots

Carpeta de salida...

Tareas

Tipo	Proyecto	Estado	Inicio	Comando	
pipeline	FacturasDemo	ok	2026-09-16 18:01:47	I:\LOCALAPPDATA\Programs\UltraGX\ultragx-engine.exe pipeline --source xpt --out I:\modelos\UltraGX\dist\manual-shots\FacturasDemo --lang es --input	Log

Pestaña Tareas

Cada pipeline y cada narrativa que lanzás queda como una tarea con su **Tipo**, **Estado**, **Inicio** y **Comando**. **Log** muestra la salida completa de la tarea (también de las que ya terminaron, mientras la aplicación siga abierta) y **Detener** la corta. Se puede ejecutar un pipeline y una narrativa a la vez, pero no dos pipelines ni dos narrativas simultáneas.

No cierres la aplicación con tareas en curso: se pierde el seguimiento y el log. Esperá a que terminen o detenelas antes. Una narrativa detenida se puede **continuar** después sin repetir los objetos ya procesados.

Configuración

UltraGX

Proyectos

Visor

Ejecutar pipeline

Narrativa

Tareas

Configuración

Configuración

Idioma de la interfaz

Español

Carpeta de salida (una subcarpeta por ejecución)

I:\models\UltraGX\dist>manual-shots

Examinar...

Guardar

Licencia

Estado

válida

Edición

completa

Titular

Mi Empresa S.A. <sistemas@miempresa.com>

Vence

2036-09-04

Archivo de licencia

%APPDATA%\UltraGX\ultragx.lic

Id de esta máquina

3f9c2a71b8e4d05c (envíalo al propietario para obtener una licencia atada a esta máquina)

Código de activación (DEMO-XXXX / ULG)

URL del servicio de licencias (opcional)

Activar con un código

Instalar archivo de licencia...

Entorno

Aplicación

0.2.7 · Microsoft Windows 10.0.26100

Motor

compilado: %LOCALAPPDATA%\Programs\UltraGX\ultragx-engine.exe (UltraGX engine 0.2.7 (pack rel-2026.09))

Conector GeneXus

%LOCALAPPDATA%\Programs\UltraGX\connector\UltraGX.Gx.Gateway.exe

GeneXus instalado

18u5 (18.0.5.44509), 18 (18.0.15.57673), 18u10 (18.0.10.53188), 17 (17.0.11.32605), 16 (16.0.11.15999), 15 (15.0.4.48249), XEv3 (10.3.15.50288), XEv2 (10.2.7.25097)

Carpeta de salida

ok

Archivo de configuración

%APPDATA%\UltraGX/settings.json (or ~/.config/UltraGX)

Pestaña Configuración

- **Idioma de la interfaz:** English, Español o Português (BR). No cambia el idioma de los documentos, que se elige en cada ejecución.
- **Carpeta de salida:** la misma que muestra la barra superior. Tocá **Guardar** después de cambiarla.
- **Licencia:** estado, activación con código e instalación de un archivo de licencia. Ver [Primer inicio y licencia](#).
- **Entorno:** lo que la aplicación detectó en la máquina.

Entorno	Qué significa
Aplicación	versión y sistema operativo
Motor	ruta y versión del motor compilado que se usa
Conector GeneXus	ruta del conector, o no encontrado (en Linux y macOS siempre; en Windows solo si se usa una copia sin conector)
GeneXus instalado	instalaciones de GeneXus encontradas, con su versión; si no hay ninguna, se trabaja con XPZ o export de texto
Carpeta de salida	ok si existe
Archivo de configuración	dónde se guardan la configuración, las conexiones y la licencia

Documentar una KB

Documentar una Knowledge Base es ejecutar el **pipeline** sobre ella: la pestaña **Ejecutar pipeline** lee la KB, la analiza y escribe la documentación completa en una subcarpeta de la carpeta de salida.

Pestaña Ejecutar pipeline

1. Elegir la fuente

Fuente	Qué se elige en Entrada	Cuándo usarla
xpz — export XPZ / XML (GeneXus 7.5 a 18)	un archivo <code>.xpz</code> o <code>.xml</code>	El caso general. Sirve para cualquier versión y en cualquier sistema operativo. La versión de GeneXus se detecta sola
gxtxt — carpeta de export de texto (src/ + ref/)	la carpeta que contiene <code>src/</code> y <code>ref/</code>	Si ya tenés un export de texto de GeneXus 17 o 18
live — abrir la KB con el GeneXus instalado en esta máquina (conector)	la carpeta de la KB (la que contiene el <code>.gxw</code>)	Windows con GeneXus X Evolution 1 a 18 instalado. No hace falta exportar nada a mano
mcp — carpeta de volcados JSON	una carpeta de volcados JSON	Uso interno y soporte; normalmente no se usa

Debajo del selector, la aplicación explica cada fuente. Si la opción **live** aparece deshabilitada (*requiere el conector y una instalación de GeneXus*), la máquina no tiene GeneXus o se está usando

una copia sin conector.

Cómo obtener un XPZ

En el IDE de GeneXus: **Knowledge Manager → Export**, marcá todos los objetos de la KB y exportá a un archivo `.xpz`. En versiones viejas (7.5, 8, 9) el camino equivalente es **Knowledge Manager → Distribute**. Incluí todo: cuanto más completo el export, mejores son las referencias, el modelo de datos y los clusters.

Consejos:

- Exportá desde la **KB de desarrollo** actualizada, no desde una copia vieja.
- Si la KB es muy grande y el IDE no puede exportarla de una vez, exportá por partes (por módulo o por tipo de objeto) y documentá cada archivo como un proyecto aparte, o usá la fuente **live**.
- Si documentaste la KB con una versión de UltraGX anterior a la 0.2.5 y los SDTs aparecen sin miembros, volvé a ejecutar el pipeline (y, con la fuente live, la exportación) con la versión actual.

Fuente live y versión de GeneXus

Con **live**, UltraGX abre la KB con el GeneXus instalado, la exporta a texto dentro de la carpeta del proyecto (`_source_text/`) y después ejecuta el pipeline. En **Versión de GeneXus para la KB en vivo** dejá **auto (desde la KB)**: el conector elige la instalación que corresponde a la versión con la que se guardó la KB. Si tenés varias instalaciones de la misma versión (por ejemplo dos *upgrades* de GeneXus 18), podés forzar una.

Antes de ejecutar con **live**:

- **Cerrá la KB en el IDE de GeneXus.** Una KB solo la puede tener abierta un proceso a la vez; si el IDE (u otra herramienta) la tiene abierta, la exportación espera o falla.
- Contá con que la exportación en vivo es la etapa más lenta: el conector extrae cada objeto por el SDK de GeneXus. Una KB de 5.000 objetos puede tardar más de una hora solo en exportar. El log va mostrando qué objeto está extrayendo.

2. Nombre del proyecto e idioma

- **Nombre del proyecto (subcarpeta de salida):** por defecto, el nombre del archivo o carpeta de entrada. Si ejecutás de nuevo con el mismo nombre, el proyecto se regenera en la misma carpeta y se conservan las **notas del usuario** y la **narrativa** ya generada.
- **Idioma de los documentos:** English, Español o Português (BR). Es el idioma de la prosa de los documentos y del visor. El `README.md` de la raíz de la salida siempre queda en inglés, porque está pensado para agentes de IA.
- **gxttext: omitir módulos referenciados (ref/):** con las fuentes gxttext y live, deja afuera los objetos de módulos que la KB solo referencia (librerías de GeneXus y de terceros). Acelera la ejecución y limpia el inventario; la contracara es que las llamadas a esos módulos quedan como referencias externas.

3. Ejecutar

Hacé clic en **Ejecutar**. El log de abajo muestra el avance por etapas:

```
[ 0.0s] reading the xpz source
[ 0.4s] reading 1/2219 Main Programs
[ 0.8s] reading 2219/2219 WorkWithPlusFactura
[ 1.8s] analysing 493/696 SDT.LinkList
[ 2.8s] analysing 578/696 WebPanel.GAMChangeRepository
[ 3.9s] analysing 603/696 WebPanel.GAMEventSubscriptionEntry
[ 4.6s] analysing 696/696 WorkWithPlusInstance.WorkWithPlusFactura
[ 4.6s] extracted 696 objects
[ 4.6s] semantic rules and validation on 696 objects
[ 5.5s] indexing the KB
[ 5.5s] resolving references and building the graph
[ 5.8s] clustering
[ 5.8s] rebuild plan per cluster
[ 5.9s] cluster knowledge and functional flow
[ 6.5s] writing 696 object documents
[ 6.5s] data model (foreign keys and ERDs)
[ 6.5s] writing 1/696 Procedure.GoogleDirectionsServiceRequest
```

Log de una ejecución

- **Lectura:** el motor lee la fuente y nombra el objeto que está procesando.
- **Análisis:** reglas, eventos, variables, accesos a datos y referencias de cada objeto.
- **Grafo, clusters, conocimiento y plan de reconstrucción:** relaciones entre objetos.
- **Escritura:** documentos, modelo de datos, diagramas y visor.
- **Aceptación:** control de calidad final.

Para no inundar la pantalla, el log escribe como mucho una línea por segundo por etapa. Si una línea se repite con el mismo objeto durante un rato, el motor sigue trabajando en un objeto grande: no está colgado.

Al terminar bien, el estado pasa a **listo**, el proyecto aparece en **Proyectos** y se abre solo en el **Visor**. Si la ejecución falla, el estado dice **terminó con estado error** y el log muestra el motivo (ver [Solución de problemas](#)).

Tiempos de referencia

KB	Fuente	Tiempo aproximado
700 objetos	XPZ	menos de un minuto
5.000 objetos	XPZ o export de texto	de algunos minutos a media hora
5.000 objetos	live	la exportación (una hora o más) más el pipeline

Los tiempos dependen del disco y del tamaño del código de cada objeto. Con la demo se suma una consulta breve al servicio de licencias.

Entradas recientes

Entradas recientes guarda las últimas quince entradas usadas. Hacé clic en una para volver a cargarla en **Entrada**.

Abrir un XPZ con doble clic

Si al instalar marcaste *Abrir exports de GeneXus (.xpz) con UltraGX*, el doble clic sobre un `.xpz` abre la aplicación en **Ejecutar pipeline** con la fuente **xpz** y el archivo ya cargado: solo falta revisar el nombre del proyecto y el idioma, y hacer clic en **Ejecutar**.

Leer el resultado

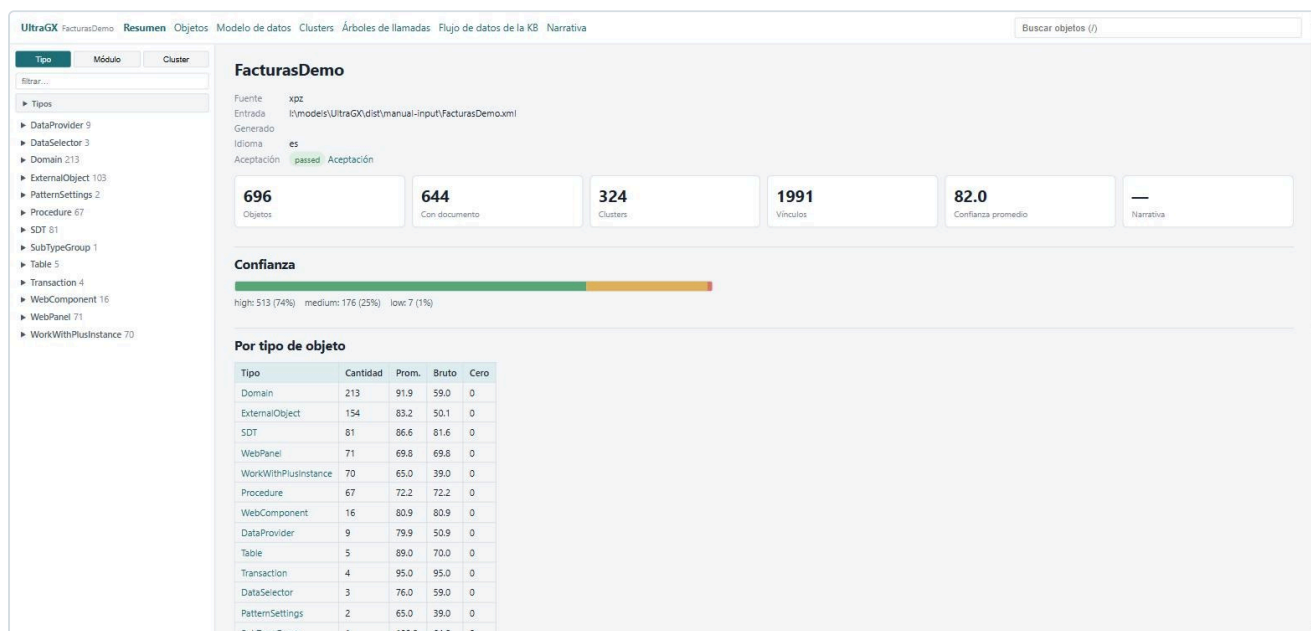
Cada ejecución deja dos cosas: un **visor web** para leer y navegar, y una **carpeta de archivos** Markdown y JSON para procesar, versionar o entregarle a un agente de IA. Los dos tienen el mismo contenido.

El visor

El visor se abre en la pestaña **Visor** de la aplicación o con **Abrir en el navegador**. Es un sitio estático que está en `viewer/` dentro del proyecto: funciona sin conexión y se puede copiar a otra máquina junto con la carpeta.

La barra superior tiene las secciones principales y el buscador (**Buscar objetos**, atajo de teclado `/`), que encuentra objetos por nombre desde cualquier página.

Resumen



Resumen del proyecto

La portada muestra de dónde salió la KB (fuente, entrada, fecha, idioma), el resultado del control de calidad y las cifras generales: objetos, objetos con documento, clusters, relaciones, confianza media y estado de la narrativa. Debajo están la distribución de la **confianza**, la tabla por tipo de objeto, los clusters más grandes y los enlaces a los informes de la KB (resumen, hoja de ruta de reconstrucción, validación, aceptación).

Confianza (0 a 100) mide cuánta evidencia encontró el motor para describir cada objeto: `high`, `medium` o `low`. Una confianza baja no significa que el objeto esté mal, sino que tiene poca estructura para analizar (por ejemplo, un procedimiento vacío o un objeto de configuración).

Objetos

UltraGXFacturaDemoResumenObjetosModelo de datosClustersÁrboles de llamadasFlujo de datos de la KBNarrativa

Buscar objetos (/)

TipoMóduloCluster

Filtrar...

Tipos

►DataProvider 9

►DataSelector 3

►Domain 213

►ExternalObject 103

►PatternSettings 2

►Procedure 67

►SDT 81

►SubTypeGroup 1

►Table 5

►Transaction 4

►WebComponent 16

►WebPanel 71

►WorkWithPlusInstance 70

Objetos

filtrar...TipoMóduloCarpetaPropio / LibreríaConfianzaNarrativa645 / 645

0 seleccionadosLimpiar

Escribi una nota sobre este objeto...

AutorAgrega nota a los objetos seleccionados

Objetos	Tipo	Módulo	Carpeta	Cluster	Perfil	Confianza	Narrativa	Notas		
GetHomeModulesSample	DataProvider	WWPBaseObjects		Librería	cluster_014	simple_provider	83	high	pending	
GetHomeSampleData	DataProvider	WWPBaseObjects		Librería	cluster_002	simple_provider	77	high	pending	
GetHomeSampleDataService	DataProvider	WWPBaseObjects		Librería	cluster_002	simple_provider	77	high	pending	
GetHomeSampleNameValueData	DataProvider	WWPBaseObjects		Librería	cluster_002	simple_provider	77	high	pending	
GetMainHomeModulesSample	DataProvider	WWPBaseObjects		Librería	cluster_014	simple_provider	77	high	pending	
GetNotificationSamples	DataProvider	WWPBaseObjects		Librería	cluster_002	simple_provider	83	high	pending	
GetSomeHomeModulesSample	DataProvider	WWPBaseObjects		Librería	cluster_014	simple_provider	77	high	pending	
MenuOptionsData	DataProvider	WWPBaseObjects		Librería	cluster_002	simple_provider	91	high	pending	
SecGAMGetAdvancedSecurityWWPFunctionalities	DataProvider	WWPBaseObjects		Librería	cluster_002	simple_provider	77	high	pending	
ClienteWWDS	DataSelector	Root Module	Propio	cluster_009	parameterized_selector	76	high	pending		
FacturaWWDS	DataSelector	Root Module	Propio	cluster_007	parameterized_selector	76	high	pending		
ProductoWWDS	DataSelector	Root Module	Propio	cluster_010	parameterized_selector	76	high	pending		
APIAuthorizationStatus	Domain	GeneXus		Librería	cluster_169	enumerated_domain	100	high	pending	
ActionTexts	Domain	Root Module	Propio	cluster_001	enumerated_domain	100	high	pending		
Address	Domain	GeneXus		Librería	cluster_006	base_domain	92	high	pending	
ApplicationState	Domain	GeneXus		Librería	cluster_146	enumerated_domain	100	high	pending	
AudioAPISessionType	Domain	GeneXus		Librería	cluster_173	enumerated_domain	100	high	pending	
BarcodeType	Domain	GeneXus		Librería	cluster_179	enumerated_domain	100	high	pending	
BeaconProximity	Domain	GeneXus		Librería	cluster_012	enumerated_domain	100	high	pending	
CUIT	Domain	Root Module	Propio	cluster_001	validated_domain	92	high	pending		
CallTargetSize	Domain	GeneXus		Librería	cluster_171	enumerated_domain	100	high	pending	

Lista de objetos

La lista de todos los objetos con filtros por texto, **Tipo** (incluida la opción **Objetos de negocio**, que deja Transacciones, Web Panels, Work Panels, Procedimientos, Reportes, Query Objects y SDTs), **Módulo**, **Carpeta**, **Propio / Librería**, **Confianza** y estado de **Narrativa**. Sobre la lista hay un cuadro para agregar una nota a los objetos seleccionados (ver **Notas del usuario**).

La **barra lateral**, presente en todas las páginas, agrupa los objetos por **Tipo**, **Módulo**, **Ubicación** (el mismo árbol de módulos y carpetas del *KB Explorer* de GeneXus, cuando la KB tiene carpetas) o **Cluster**. En **Tipos** se eligen qué tipos mostrar y, si la KB tiene objetos de módulos importados, **Ocultar objetos de librería** los deja afuera.

Página de un objeto

UltraGXFacturaDemoResumenObjetosModelo de datosClustersÁrboles de llamadasFlujo de datos de la KBNarrativa

Buscar objetos (/)

TipoMóduloCluster

Filtrar...

Tipos

►DataProvider 9

►DataSelector 3

►Domain 213

►ExternalObject 103

►PatternSettings 2

►Procedure 67

►SDT 81

►SubTypeGroup 1

►Table 5

▼Transaction 4

Cliente

Factura

Producto

UserCustomizations

►WebComponent 16

►WebPanel 71

►WorkWithPlusInstance 70

TransactionMódulo: Root Module Perfil: multilevel_transaction Acceso a datos: multilevel_transaction Confianza: 100/100 high Cluster: cluster_022 - mixed_module - data_core Narrativa: pendiente

Árbol de llamadasFlujo de datosClusterKnowledge packModelo de datosFuente .mdAnálisis .json

Usa (8)

- Cliente data
- FacturaLoadDV/Combo nominal_reference
- FacturaWW call
- LoadWWPContext call
- ProductoPrompt nominal_reference
- SecGAMFunctionalityKey data
- WWPContext data
- WWPTransactionContext data

Usado por (11)

- ClienteFacturaWC call
- ClienteView data
- FacturaFacWC nominal_reference
- FacturaGeneral call
- FacturaLoadDV/Combo candidate
- FacturaPrompt data
- FacturaWW call
- FacturaWWExportReport data
- ListWWPPrograms data
- ProductoFacFacturaWC call
- WorkWithPlusFactura parent

► Notas del usuario 0

Transaction Factura

- Módulo / carpeta: Root Module
- Perfil funcional: multilevel_transaction - acceso a datos: multilevel_transaction
- Confianza del análisis: 100/100 (high) - fuente: mcp
- Generado o gobernado por un pattern (WorkWith / K28): parte del código no está escrito a mano.

Qué hace

- Factura es una Transaction: define y mantiene (alta, modificación, baja) la entidad **Factura**, persistida en la tabla **Factura** con 13 atributos; clave primaria: **FacId**.
- Tiene un nivel de detalle **Fac** (8 atributos, clave **FacSec**) guardado en su propia tabla **FacturaFac**: un **Factura** tiene muchas filas **Fac**.
- Referencia a **Producto** a través de **prosta** (de allí lee **prodmon**, **prodprc**, **prodstk**, **prodsts**, que no se guardan acá).
- Contrato**: recibe **anode**, **afac16** (AttributeFacId (Id (Numeric(12)))).
- anode** elige la operación (Insert / Update / Delete / Display) y la clave llega por parámetro: se abre desde otra pantalla para un registro concreto.
- Aplica 5 validación(es) con mensajes de error antes de grabar (ver reglas de negocio).
- Al abrir la pantalla (Start) llama a **LoadWWPContext**, oculta **FacC11id**, **&ComboFacC11id**, **FacId**, **FacU1Sec**.
- Desde **de grabar con datos** (XFac - tes 3) llama a **FacturaWW** cuando **&Mode = Transaction** (data and not .K28Transaction) ... usa los xl llamador

Documento de un objeto

Arriba están la identidad del objeto (tipo, módulo, carpeta, cluster, perfil funcional, acceso a datos, confianza), sus relaciones (**Usa** y **Usado por**, con enlaces) y las **Notas del usuario**. Después viene el documento, con el índice de secciones a la derecha:

Sección	Qué contiene
Resumen	tabla con los datos clave del objeto
Qué hace	descripción en prosa de su propósito
Modelo de datos	(transacciones) niveles, atributos con su rol (clave, clave foránea, fórmula, inferido) y diagrama entidad-relación
Reglas de negocio	cada regla, explicada una por una
Comportamiento (pseudocódigo)	lo que hace cada evento y subrutina, en pseudocódigo legible
SQL implícito (por For each)	la consulta que GeneXus genera en cada For each : tabla base, joins, condiciones y orden
Parámetros y variables	la regla <code>parm</code> y las variables con su tipo
Interfaz de usuario	campos, grillas y su tabla base, botones y acciones, comportamiento dinámico
Significado de negocio	prosa de la narrativa con IA, si se generó
Referencias cruzadas	objetos relacionados
Límites de este análisis	lo que el motor no pudo determinar con certeza
Código GeneXus original (literal)	el código fuente tal cual está en la KB

Al final, desplegable, está el **Anexo determinista del motor**: la evidencia técnica completa del análisis (hallazgos, riesgos, puntaje, trazabilidad). Sirve para auditar una conclusión; para leer el objeto no hace falta.

Las secciones cambian según el tipo: un Data Provider muestra su salida, un SDT su estructura, un Domain sus valores, etc.

Modelo de datos

Modelo de datos de la Transaction Factura

- Factura · Módulo: Root Module · volver al modelo de datos

Niveles y tablas

Nivel	Tabla	Claves	Columnas	Referencia a
Factura	Factura	FacId	13	Ciente
Fac	FacturaFac	FacId, FacSec	5	Factura, Producto

Diagrama entidad-relación

El diagrama entidad-relación muestra la siguiente estructura:

- Entidad Cliente:** Clave primaria `Id_Numeric_12`. Clave foránea `FacCild` que se relaciona con la clave foránea `FacCild` de la entidad Factura.
- Entidad Factura:** Clave primaria `Id_Numeric_12`. Claves foráneas: `FacCild` (FK a Cliente), `FacCildNom` (FK a Cliente), `FacUpd` (FK a Cliente), `FacUpdUsr` (FK a Cliente), `FacSts` (FK a Cliente), `FacUltSec` (FK a Cliente), `FacTotal` (FK a Cliente).
- Entidad Producto:** Clave primaria `Id_Numeric_12`. Clave foránea `Prodid` que se relaciona con la clave foránea `Prodid` de la entidad Factura.

Modelo de datos

El diagrama entidad-relación de toda la KB, la lista de tablas y el detalle por transacción. Cada tabla tiene su página con columnas (clave primaria y foráneas), relaciones, transacciones que la definen, objetos que la usan y un DDL conceptual.

Clusters

Un **cluster** es un grupo de objetos que trabajan juntos (por ejemplo, todo lo relacionado con facturación). El motor los arma a partir del grafo de dependencias y de la ubicación en la KB. Cada cluster tiene su clase, prioridad, objeto núcleo, puntos de entrada y miembros, y tres informes: **Conocimiento** (propósito, datos, riesgos), **Flujo funcional** (cómo pasa el trabajo de un objeto a otro) y **Plan de reconstrucción**.

La **Hoja de ruta de reconstrucción** (desde el Resumen) ordena los clusters en fases según sus dependencias: primero lo que no depende de nada, después lo que se apoya en eso.

Árboles de llamadas y flujo de datos

- Árboles de llamadas:** desde cada punto de entrada, quién llama a quién. Los nodos del diagrama son clicables.
- Flujo de datos de la KB:** qué objetos leen y escriben qué tablas, para toda la KB, por cluster y por objeto.

Los diagramas muy grandes no se dibujan automáticamente: aparece el botón **Dibujar diagrama**, y también se puede ver el texto con **Fuente del diagrama**.

Narrativa

Estado de la narrativa con IA por objeto: `ok`, `degraded`, `error` o pendiente. Ver [Narrativa con IA](#).

La carpeta de salida

Todo lo que muestra el visor está en archivos dentro de la carpeta del proyecto:

Ruta	Contenido
<code>README.md</code>	Punto de entrada: cómo usar la carpeta, mapa, glosario, clusters, tablas e inventario completo. Siempre en inglés
<code>run_summary.json</code>	Cómo se generó: fuente, entrada, idioma, cantidades, licencia, aceptación
<code>kb_index.json</code>	Una fila por objeto: tipo, módulo, carpeta, perfil, confianza, cluster y <code>doc_path</code> , la ruta exacta de su documento
<code>kb_graph.json</code>	Grafo de dependencias: relaciones entrantes y salientes de cada objeto
<code>kb_summary.md</code> / <code>.json</code>	Métricas de la KB
<code>kb_clusters.json</code> , <code>kb_clusters_summary.md</code>	Clusters
<code>global_rebuild_roadmap.md</code> / <code>.json</code>	Hoja de ruta de reconstrucción por fases
<code>acceptance_report.md</code> / <code>.json</code>	Control de calidad final
<code>objects/</code>	Un <code><Tipo>.<Nombre>.md</code> y un <code>.json</code> por objeto, en carpetas que replican módulos y carpetas de la KB
<code>knowledge/</code>	Knowledge pack de cada objeto
<code>datamodel/</code>	Modelo de datos: índice, una página por tabla y por transacción, <code>kb_datamodel.json</code>
<code>clusters/</code>	Conocimiento, flujo y plan de reconstrucción de cada cluster
<code>calltree/</code> , <code>dfd/</code>	Árboles de llamadas y diagramas de flujo de datos
<code>narrative/</code>	Resultado de la narrativa con IA, un JSON por objeto
<code>annotations/</code>	Notas del usuario
<code>viewer/</code>	El visor. Se regenera: no lo edites a mano
<code>_pipeline/</code> , <code>_source_text/</code>	Uso interno: instantáneas para detectar cambios y, con la fuente live, el export de texto

Los documentos son Markdown estándar con diagramas Mermaid: se leen bien en GitHub, GitLab, VS Code u Obsidian.

Narrativa con IA

El pipeline documenta la KB con reglas deterministas. La **narrativa** es un paso opcional que agrega, a cada documento, prosa escrita por un modelo de lenguaje: el propósito del objeto, un resumen, la intención de cada evento, las reglas explicadas en términos de negocio, notas para la migración y un glosario.

Lo que la distingue de pegarle el código a un chat es la **validación**: el modelo recibe el documento ya analizado (no el GeneXus crudo) y UltraGX verifica que cada atributo, tabla, variable u objeto que cite exista realmente en ese objeto. Si el modelo inventa un nombre, se le pide que corrija; si insiste, la prosa no se publica.

La narrativa se ejecuta sobre un proyecto **ya generado** por el pipeline.

The screenshot shows the 'Narrativa' tab in the UltraGX application. The title is 'Narrativa (prosa de LLM validada contra los hechos)'. The form contains the following fields and controls:

- Proyecto:** A dropdown menu with 'FacturasDemo' selected.
- Conexión:** A dropdown menu with 'Gemini (empresa)' selected. Below it, there is a text input with 'Gemini (empresa)', a '+ Nueva' button, and a 'Borrar' button.
- Proveedor:** A dropdown menu with 'Google (Gemini)' selected.
- Modelo:** A text input with 'gemini-3.8-flash'.
- URL base (compatible con OpenAI):** A text input with 'https://generativelanguage.googleapis.com/v1beta/openai/'.
- API key de esta conexión (se guarda en el archivo de configuración):** A text input with a masked key '*****'.
- Idioma de los documentos:** A dropdown menu with 'Español' selected.
- Concurrencia:** A text input with '4'.
- Tipos (separados por coma, vacío = todos):** A text input with 'procedure,transaction,webpanel'.
- Límite (0 = todos):** A text input with '0'.
- Caracteres del documento que se mandan al modelo (bajalo a 16.000 para un modelo local de contexto chico):** A text input with '60000'.

Pestaña Narrativa

Conexiones

Una **conexión** es un proveedor, un modelo y sus credenciales, guardados con un nombre. Podés tener varias (por ejemplo, una cuenta de la empresa y un modelo local) y elegir cuál usar en cada ejecución.

1. En **Conexión**, hacé clic en **+ Nueva** y escribí un nombre.

- 2. Elegí el **Proveedor**. El **Modelo** y la **URL base** se completan con valores por defecto que podés cambiar.
- 3. Pegá la **API key de esta conexión**, o marcá **Corre local** si usás un modelo en tu equipo o tu red.
- 4. Los cambios se guardan al ejecutar. **Borrar** elimina la conexión seleccionada.

Proveedor	Modelo por defecto	Notas
Anthropic (Claude)	claude-sonnet-5	Muy buena calidad de prosa y de respeto a los nombres
OpenAI (GPT)	gpt-4o-mini	
Google (Gemini)	gemini-3.8-flash	Contexto amplio y rápido; buena opción para KB grandes
DeepSeek, Zhipu (GLM), Moonshot (Kimi), Alibaba (Qwen)	según el proveedor	Admiten Corre local si el modelo está en tu propia infraestructura
Google (Gemma)	gemma3:latest	Pensado para correr local
OpenRouter	un modelo de su catálogo	Una sola clave para muchos modelos
mock (sin modelo)	—	Genera texto de prueba, sin costo; sirve para verificar el flujo

Modelos locales

Con **Corre local**, la API key se oculta y la **URL base** apunta a tu servidor compatible con OpenAI: `http://localhost:11434/v1` para Ollama, `http://localhost:1234/v1` para LM Studio, o la dirección de otra máquina de tu red. Instalá y descargá el modelo en ese servidor antes de ejecutar (por ejemplo `ollama pull qwen3.6`).

Con un modelo local **ningún documento sale de tu red**. La contracara es la velocidad y el tamaño de contexto: si el modelo tiene una ventana de contexto chica, bajá **Caracteres del documento que se mandan al modelo** a 16.000 o menos.

Dónde se guardan las claves

Las API keys se guardan en el archivo de configuración de UltraGX (`%APPDATA%\UltraGX\settings.json` en Windows, `~/.config/UltraGX/settings.json` en Linux y macOS), dentro de tu perfil de usuario. No se escriben en la carpeta de salida ni se envían a UltraGX. Tratá ese archivo como un secreto.

Opciones de la ejecución

Opción	Qué hace
Proyecto	el proyecto a narrar
Idioma	idioma de la prosa (conviene el mismo del proyecto)
Concurrencia	objetos en paralelo. Con proveedores en la nube, 4 a 8 acelera mucho; con un modelo local, 1

Opción	Qué hace
Tipos	tipos de objeto a narrar, separados por coma (por ejemplo <code>procedure, transaction, webpanel</code>). Vacío = todos los tipos con lógica
Límite	cantidad máxima de objetos en esta ejecución (0 = todos). Útil para probar con 10 objetos antes de lanzar la KB entera
Caracteres del documento que se mandan al modelo	tamaño máximo de cada documento enviado (60.000 por defecto). Los documentos más largos se recortan

Con **Tipos** vacío, UltraGX **no narra las declaraciones**: atributos, dominios, SDTs, grupos de subtipos, objetos externos, data views y configuraciones de patrones. Su documento determinista ya dice todo lo que hay que decir y narrarlas solo gasta tokens. Si igual querés narrar alguno, escribilo en **Tipos**.

Ejecutar, seguir y continuar

Hacé clic en **Ejecutar / continuar**. El log muestra cada objeto con su resultado. Cada respuesta se incorpora a su documento **en el momento**: podés abrir el objeto en el **Visor** (con **Recargar**) mientras la ejecución sigue, y el visor completo se regenera periódicamente y al final.

Si detenés la ejecución o se corta, volvé a hacer clic en **Ejecutar / continuar**: los objetos ya narrados no se vuelven a enviar.

Resultados por objeto

Estado	Qué significa	Se publica
ok	la prosa pasó la validación	sí, en la sección Significado de negocio y en las secciones correspondientes
degraded	después de los reintentos, el modelo siguió citando nombres que no existen en el objeto	no; el documento muestra un aviso con los nombres inventados
error	el proveedor falló (clave inválida, cuota agotada, respuesta inválida, tiempo de espera)	no

Reintentar degradados / errores vuelve a procesar solo esos objetos. Los errores transitorios del proveedor (límites de tasa, errores 5xx, cortes de red) ya se reintentan solos con espera progresiva durante la ejecución.

El estado de todos los objetos está en la página **Narrativa** del visor y en la columna **Narrativa** de la lista de objetos.

Costos y privacidad

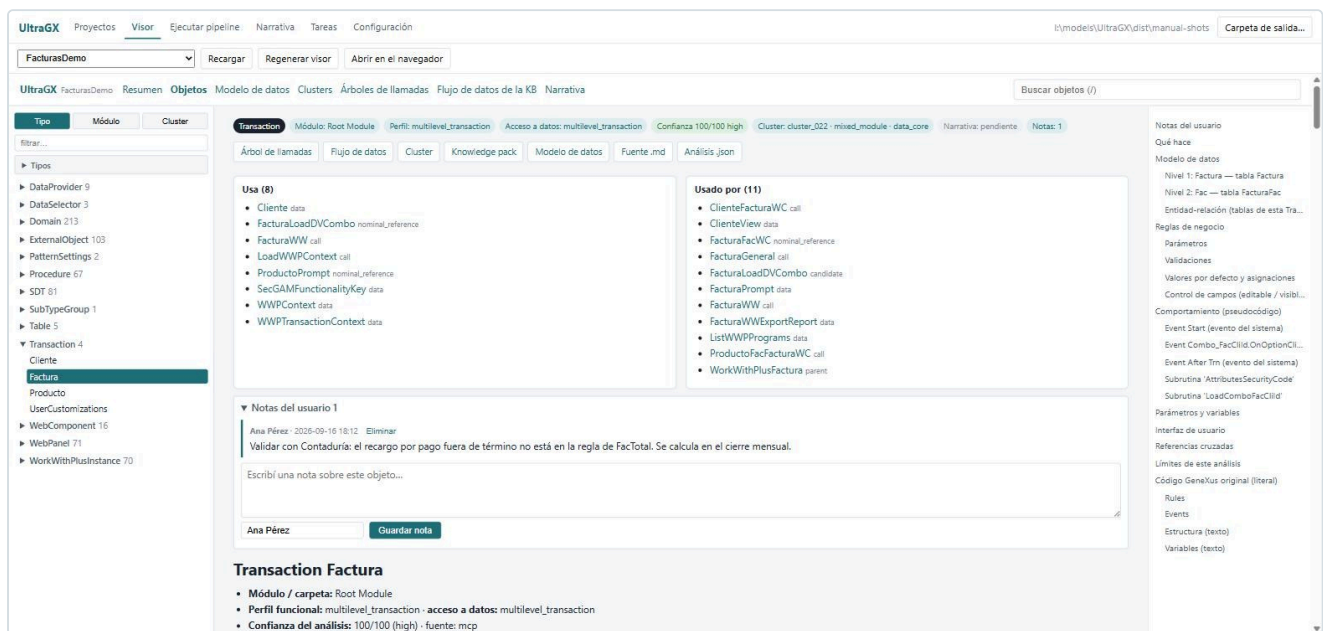
- Cada objeto es una llamada al modelo con su documento completo (hasta el límite de caracteres). Para estimar el costo, ejecutá con **Límite** 20, mirá los tokens que informa el log al final y extrapolá a la cantidad de objetos con lógica de tu KB.
- Con proveedores en la nube, el documento de cada objeto (incluido su código GeneXus) se envía a ese proveedor con tu cuenta y bajo sus condiciones. Si la KB tiene restricciones de confidencialidad, usá un modelo local o un proveedor aprobado por tu organización.
- En la edición demo solo se narran los objetos de la muestra.

Notas del usuario

Las notas guardan lo que el análisis no puede saber: que un procedimiento ya no se usa, que una regla responde a una normativa, quién es el referente funcional de un módulo o qué decidió el equipo para la migración. Quedan dentro de la documentación, así que también las lee un agente de IA.

Las notas se agregan **desde la aplicación** (pestaña **Visor**). Si abris el visor en un navegador aparte, se ven pero no se pueden crear ni borrar.

Nota sobre un objeto



Notas en la página de un objeto

1. Abrí el objeto en el **Visor**.
2. En **Notas del usuario**, escribí la nota y, si querés, tu nombre en **Autor**.
3. Hacé clic en **Guardar nota**.

La nota aparece enseguida en la página y queda con fecha y autor. **Eliminar** la borra.

Nota sobre varios objetos a la vez

1. En **Objetos**, filtrá la lista (por ejemplo, por carpeta o por cluster).
2. Marcá las casillas de la última columna, o la casilla del encabezado para **Seleccionar visibles**.
3. En la barra que aparece arriba, escribí la nota y hacé clic en **Agregar nota a los objetos seleccionados**.

Cada objeto recibe su copia de la nota. La columna **Notas** de la lista marca qué objetos tienen notas.

Dónde quedan

- En `annotations/` dentro del proyecto, un archivo JSON por objeto. Es la fuente de verdad: copíala esa carpeta si querés llevar las notas a otra máquina.
- En el documento Markdown del objeto, sección **Notas del usuario**.
- En el JSON del objeto, campo `user_notes`.

Cuando volvéis a ejecutar el pipeline sobre el mismo proyecto (por ejemplo, con un export más nuevo de la KB), las notas se vuelven a incorporar a los documentos regenerados. Si un objeto desapareció de la KB, su archivo de notas queda en `annotations/` pero ya no se muestra.

Migrar con un agente de IA

La salida de UltraGX está escrita para que un agente de programación (Claude Code, Codex, Cursor, Antigravity u otro) pueda reconstruir la aplicación en otra tecnología sin abrir GeneXus. El agente no tiene que interpretar código GeneXus desde cero: lee reglas ya explicadas, pseudocódigo por evento, el SQL implícito de cada `For each`, el modelo de datos con sus claves y un orden de reconstrucción.

Este capítulo describe un método de trabajo que funciona bien. No es la única forma, pero evita los errores más comunes: pedirle todo de una vez al agente, o dejar que invente estructura que la KB ya define.

1. Preparar el espacio de trabajo

1. Generará la documentación con la **edición completa** y, si podés, con la narrativa en `ok` para los objetos de negocio. Usá el idioma en que trabaja tu equipo.
2. Creá el repositorio de la aplicación nueva **separado** de la salida de UltraGX.
3. Dejá la salida accesible para el agente como material de solo lectura: por ejemplo, una carpeta hermana (`../kb-docs/`) o un submódulo. No hace falta copiar `viewer/`, `_pipeline/` ni `_source_text/`.
4. En el repositorio nuevo, escribí el archivo de instrucciones del agente (`CLAUDE.md`, `AGENTS.md` o las reglas de Cursor) con:
 - la tecnología de destino y sus convenciones (lenguaje, framework, base de datos, estructura de carpetas);
 - la ubicación de la documentación de la KB y la indicación de leer primero su `README.md`;
 - la regla de que los nombres de tablas, atributos y reglas salen de la documentación, no se inventan;
 - que las **notas del usuario** prevalecen sobre lo inferido.

Ejemplo de bloque para ese archivo (si tus documentos están en español, las secciones se llaman **Notas del usuario** y **Límites de este análisis**):

```
## Legacy system knowledge
```

```
The GeneXus application being migrated is documented in ../kb-docs (UltraGX output, read-only).
```

```
Start with ../kb-docs/README.md. Locate objects through kb_index.json (doc_path column).
```

```
Table and column names, keys and business rules come from those documents; never invent them.
```

```
"User notes" sections override inferred statements. When a document lists something under "Limits of this analysis", ask before assuming.
```

2. Orden de lectura

El `README.md` de la salida le indica al agente cómo recorrerla. En resumen:

1. `README.md`, `kb_summary.md` y `kb_clusters_summary.md`: qué hay y cómo se agrupa.
2. `datamodel/index.md`: tablas, claves y qué transacción define cada tabla.
3. `global_rebuild_roadmap.md`: en qué orden reconstruir.
4. Por cada cluster: `clusters/cluster_NNN.knowledge.md`, `.flow.md` y `.rebuild.md`.
5. Por cada objeto: su documento en `objects/` (ruta en `kb_index.json` → `doc_path`).
6. Para seguir relaciones: `kb_graph.json` o la sección **Referencias cruzadas** de cada documento.

3. Trabajar por fases y por cluster

1. **Modelo de datos primero.** Pedile al agente el esquema de base de datos y las entidades a partir de `datamodel/`. Revisalo contra la base real antes de seguir: todo lo demás se apoya en esto.
2. **Seguí la hoja de ruta.** Tomá la primera fase de `global_rebuild_roadmap.md` y, dentro de ella, un cluster por vez.
3. **Un cluster, una tarea.** Para cada cluster, el agente lee su plan de reconstrucción y los documentos de sus objetos, propone el diseño, lo implementa y escribe pruebas a partir de las reglas de negocio.
4. **Cerrá el cluster** antes de pasar al siguiente: pruebas en verde, revisión de las reglas una por una y notas del usuario con las decisiones tomadas (quedan para las próximas sesiones del agente).

4. Pedidos de ejemplo

Orientación

Leé `../kb-docs/README.md`, `kb_summary.md` y `kb_clusters_summary.md`. Resumime qué hace la aplicación, sus áreas funcionales y las 10 tablas más importantes. No escribas código todavía.

Modelo de datos

A partir de `../kb-docs/datamodel/`, generá las migraciones de base de datos para PostgreSQL y las entidades del ORM. Respetá nombres, tipos y claves primarias y foráneas. Listá cualquier tabla cuya definición esté incompleta.

Un cluster

Implementá el cluster `cluster_004` según `clusters/cluster_004.rebuild.md`. Leé el documento de cada objeto miembro y reproducí sus reglas de negocio y el SQL implícito de sus For each. Para las pantallas, usá la sección de interfaz de usuario. Escribí una prueba por regla de negocio y marcá lo que no puedas reproducir con certeza.

Un objeto puntual

Buscá en `kb_index.json` el procedure `CalcularDescuento`, leé su documento y reimplementalo como servicio. Explicá

cada rama del pseudocódigo en un comentario de la prueba correspondiente.

Verificación

Comparará el servicio InvoiceService con objects/Facturacion/Transaction.Factura.md: enumerará cada regla de negocio del documento e indicará dónde está implementada o si falta.

5. Buenas prácticas

- **Contexto acotado.** No le pidas al agente que lea toda la carpeta: con `kb_index.json` y `doc_path` encuentra lo que necesita para cada tarea.
- **Confianza.** Priorizá la revisión humana de los objetos con `confidence_level` bajo y de lo que aparece en **Límites de este análisis**.
- **Objetos de librería y de patrones.** Los objetos marcados como librería (módulos importados de GeneXus o de terceros) y el andamiaje de patrones (Work With, K2BTools) no se migran tal cual: se reemplazan por los equivalentes del framework de destino.
- **La KB sigue viva.** Si la aplicación GeneXus recibe cambios durante la migración, volvé a exportar y a ejecutar el pipeline sobre el mismo proyecto; las notas se conservan y las diferencias quedan visibles en el control de versiones si la salida está versionada.

Línea de comandos

Todo lo que hace la aplicación lo ejecuta el motor, `ultragx-engine`, que también se puede usar directamente desde una terminal: para automatizar la documentación de varias KB, programarla en un servidor o integrarla en un proceso de integración continua. Usa la misma licencia que la aplicación.

Dónde está el motor

Sistema	Ruta
Windows (instalador)	%LOCALAPPDATA%\Programs\UltraGX\ultragx-engine.exe
Linux (deb / rpm)	/opt/ultragx/ultragx-engine
Linux (AppImage)	./UltraGX-<versión>-x86_64.AppImage engine <comando> ...
macOS	/Applications/UltraGX.app/Contents/MacOS/ultragx-engine

En los ejemplos se usa `ultragx-engine`; reemplazalo por la ruta correspondiente o agregó la carpeta al `PATH`.

Comandos

version y machine-id

```
ultragx-engine version      # versión del motor
ultragx-engine machine-id   # id de la máquina, para pedir una licencia atada a ella
```

license

```
ultragx-engine license      # estado de la licencia, en JSON
ultragx-engine license activate --code ULGX-XXXX-XXXX  # activa una licencia con un código
```

La demo también se puede pedir sin la aplicación, en dos pasos:

```
ultragx-engine license register --first-name Ana --last-name Pérez --company "Mi Empresa" \
  --email ana.perez@miempresa.com --gx-version 18 --kb-size 3000 --lang es
ultragx-engine license confirm --email ana.perez@miempresa.com --code 123456
```

pipeline

Documenta una KB desde un export.

```
ultragx-engine pipeline --source xpz --input C:/exports/MiKB.xpz --out D:/ultragx/MiKB --lang es
ultragx-engine pipeline --source gxtxt --input D:/exports/MiKB_texto --out D:/ultragx/MiKB --lang es --no-ref
```

Opción	Descripción
<code>--source</code>	<code>xpz</code> (archivo <code>.xpz</code> o <code>.xml</code>) o <code>gxttext</code> (carpeta con <code>src/</code> y <code>ref/</code>)
<code>--input</code>	archivo o carpeta de entrada
<code>--out</code>	carpeta del proyecto (se crea si no existe)
<code>--lang</code>	idioma de los documentos: <code>en</code> , <code>es</code> o <code>pt-BR</code>
<code>--no-ref</code>	con <code>gxttext</code> , omite los objetos de módulos referenciados
<code>--no-viewer</code>	no genera el visor (más rápido; se puede generar después)

El progreso se escribe en la salida de error y, al terminar, el resumen de la ejecución sale en JSON por la salida estándar.

export

Abre una KB con el GeneXus instalado (solo Windows, con el conector), la exporta a texto y ejecuta el pipeline.

```
ultragx-engine export --kb-path "C:/Models/MiKB" --kb MiKB --out D:/ultragx/MiKB --lang es
```

Opción	Descripción
<code>--kb-path</code>	carpeta de la KB (la que contiene el <code>.gxw</code>)
<code>--kb</code>	nombre corto de la KB
<code>--out</code>	carpeta del proyecto
<code>--gx-version</code>	fuerza una instalación de GeneXus (<code>18</code> , <code>17</code> , <code>XEv3</code> ...); por defecto se deduce de la KB
<code>--names</code>	exporta solo algunos objetos: nombres o selectores como <code>type:Transaction</code>
<code>--no-reference</code>	omite los módulos referenciados
<code>--skip-pipeline</code>	solo exporta, sin documentar
<code>--timeout</code>	tiempo máximo de la exportación en segundos (3600 por defecto)
<code>--lang</code> , <code>--no-viewer</code>	igual que en <code>pipeline</code>

viewer

Regenera el visor de un proyecto sin volver a analizar la KB.

```
ultragx-engine viewer --out D:/ultragx/MiKB --lang es
```

narrative

Ejecuta la narrativa con IA sobre un proyecto ya documentado. Por línea de comandos, las credenciales se toman de variables de entorno: `ANTHROPIC_API_KEY` para Anthropic, `OPENAI_API_KEY` (y `OPENAI_BASE_URL` para otro servidor) para los proveedores compatibles con OpenAI, incluidos Gemini, OpenRouter y los modelos locales.

```
export OPENAI_API_KEY=...          # en PowerShell: $env:OPENAI_API_KEY = "..."
ultragx-engine narrative run --out D:/ultragx/MiKB --provider openai \
    --base-url https://generativelanguage.googleapis.com/v1beta/openai/ --model gemini-3.8-
flash \
    --lang es --concurrency 8
```

Opción	Descripción
<code>--provider</code>	anthropic, openai (cualquier servidor compatible) o mock
<code>--model</code> , <code>--base-url</code>	modelo y URL del servidor
<code>--lang</code>	idioma de la prosa
<code>--concurrency</code>	objetos en paralelo (4 por defecto)
<code>--types</code>	solo estos tipos, separados por coma
<code>--all-types</code>	incluye también las declaraciones (atributos, dominios, SDTs...)
<code>--only</code>	solo estos objetos, por <Tipo>.<Nombre> separados por coma
<code>--limit</code>	cantidad máxima de objetos
<code>--no-resume</code>	vuelve a procesar objetos ya narrados
<code>--max-doc-chars</code>	caracteres del documento que se envían (60.000 por defecto; 16.000 con un servidor local si no se indica)
<code>--no-source</code>	no envía el código GeneXus original al modelo
<code>--timeout</code>	segundos de espera por respuesta (300 por defecto)
<code>--viewer-every</code>	segundos entre regeneraciones del visor durante la ejecución (600 por defecto; 0 = solo al final)

notes

Agrega o borra notas del usuario sin la aplicación.

```
ultragx-engine notes add      --out D:/ultragx/MiKB --json nota.json
ultragx-engine notes delete --out D:/ultragx/MiKB --stem Procedure.CalcularDescuento --id
n171234567890000
ultragx-engine notes apply   --out D:/ultragx/MiKB
```

nota.json tiene la forma {"stems": ["Procedure.CalcularDescuento"], "text": "...", "author": "Ana"}. apply vuelve a incorporar todas las notas a los documentos.

Códigos de salida

Código	Significado
0	terminó bien
1	la narrativa terminó con objetos en error

Código	Significado
2	uso incorrecto (comando u opciones inválidos) o, en <code>export</code> , no se pudo abrir la KB
3	licencia faltante, inválida, vencida o revocada

Ejemplo: documentar todas las KB de una carpeta

```
$engine = "$env:LOCALAPPDATA\Programs\UltraGX\ultragx-engine.exe"
Get-ChildItem D:\exports\*.xpz | ForEach-Object {
    & $engine pipeline --source xpz --input $_.FullName --out "D:\ultragx\${$_.BaseName}" --
lang es
    if ($LASTEXITCODE -ne 0) { Write-Warning "${$_.Name} terminó con código $LASTEXITCODE" }
}
```

Solución de problemas y preguntas frecuentes

Instalación y arranque

Windows muestra "Windows protegió su PC". Es SmartScreen con un instalador que todavía tiene poca reputación. Hacé clic en **Más información → Ejecutar de todas formas**. Si tu organización bloquea esa opción, pedile a soporte técnico que habilite el instalador.

La ventana de UltraGX se abre en blanco o no se abre (Windows). Falta el runtime Microsoft Edge WebView2 o está dañado. Instalalo desde la página de Microsoft ("WebView2 Runtime", instalador *Evergreen*) y volvé a abrir UltraGX. El instalador de UltraGX lo agrega solo.

La aplicación no abre en Linux. Ejecutala desde una terminal (`ultragx` o la ApplImage) para ver el error. Si menciona `libwebkit2gtk`, instalá `libwebkit2gtk-4.1-0` (Debian/Ubuntu) o `webkit2gtk4.1` (Fedora).

macOS dice que UltraGX "no se puede abrir". La versión de macOS no está firmada con un Developer ID de Apple. Hacé doble clic en **Allow UltraGX.command** de la imagen de disco después de arrastrar UltraGX a Aplicaciones (o ejecutá `xattr -cr /Applications/UltraGX.app` en Terminal), o autorizala en **Configuración del Sistema → Privacidad y seguridad → Abrir igualmente**.

Licencia

Aparece "No se pudo contactar al servicio de licencias". El equipo no llega a `https://licenses.ultra-gx.com`. Verificá la conexión a internet; en redes corporativas, pedí que permitan HTTPS a ese dominio (también a través del proxy). Los demás mensajes del registro están explicados en [Primer inicio y licencia](#).

La ejecución termina con "A valid license file (ultragx.lic) is required..." o código 3. La licencia falta, venció o fue revocada. Revisá **Configuración → Licencia**. Si la demo terminó, escribinos para pasar a la edición completa.

En la demo, ningún objeto salió detallado. La ejecución no pudo consultar el servicio de licencias y siguió sin muestra. El motivo está en el aviso del `README.md` y del visor. Resolvé la conexión y volvé a ejecutar el pipeline.

Cambié de computadora. La licencia queda atada a la máquina donde se activó. Escribinos con el nuevo **Id de esta máquina** para moverla.

Documentar una KB

La fuente "live" aparece deshabilitada. **Configuración → Entorno** muestra por qué: *Conector GeneXus: no encontrado* (Linux, macOS o una copia sin conector) o *GeneXus instalado: ninguno detectado*. En esos casos exportá la KB a XPZ en la máquina donde está GeneXus.

La exportación en vivo no avanza o falla al abrir la KB. La KB está abierta en el IDE de GeneXus o en otra herramienta: cerrala y volvé a ejecutar. Si el log repite el mismo objeto durante varios minutos, el conector sigue extrayendo un objeto grande; en KB de miles de objetos la exportación puede llevar más de una hora.

Elegí la versión de GeneXus equivocada. Dejá **auto (desde la KB)**. Forzá una instalación solo si la KB se abre con una versión distinta de la que se detecta.

Algunos SDTs o transacciones aparecen sin estructura. Si la KB se documentó con una versión anterior a la 0.2.5, volvé a ejecutarla con la versión actual (con la fuente live, también la exportación).

La aceptación terminó en warning. La documentación se generó completa; el control de calidad encontró algo a revisar. El detalle está en **Aceptación**, en el visor. `failed` indica un problema que conviene reportar.

El resumen avisa `library_flag_suspect`. Casi todos los objetos quedaron marcados como librería, lo que indica un export que no separó bien los objetos propios de los referenciados. Volvé a exportar la KB (con el conector actual, si es en vivo).

Las rutas o los nombres de carpeta son muy largos (Windows). La salida replica los módulos y carpetas de la KB. Usá una carpeta de salida corta, como `D:\ultragx`.

La carpeta de salida está en OneDrive / Dropbox y todo va lento. La sincronización procesa miles de archivos a la vez. Usá una carpeta fuera de las carpetas sincronizadas.

Narrativa

Mensaje en el log	Causa	Solución
<code>401</code> , <code>403</code> , <code>invalid API key</code>	la clave de la conexión no es válida o no tiene permisos para ese modelo	revisá la API key de la conexión
<code>404</code> , <code>model not found</code>	el nombre del modelo no existe en ese proveedor	corregí Modelo con un nombre del catálogo del proveedor
<code>429</code> , <code>quota</code> , <code>rate limit</code>	límite de uso del proveedor	el motor reintenta solo; si persiste, bajá Concurrencia o revisá la cuota de tu cuenta
<code>connection refused</code> con un modelo local	el servidor local no está corriendo o la URL no es la correcta	iniciá Ollama / LM Studio y verifiqué la URL base
muchos <code>degraded</code>	el modelo cita nombres que no existen	probá un modelo más capaz; con modelos locales, subí el tamaño de contexto

La prosa no aparece en el visor. Hacé clic en **Recargar**. Solo se publican los objetos en `ok`.

Dónde mirar

Qué	Dónde
Log de una ejecución	Tareas → Log mientras la aplicación está abierta
Resultado de la ejecución	<code>run_summary.json</code> y <code>acceptance_report.md</code> en la carpeta del proyecto
Estado de la narrativa	<code>narrative/checkpoint.json</code> y un JSON por objeto en <code>narrative/</code>

Qué	Dónde
Configuración y licencia	%APPDATA%\UltraGX (Windows) o ~/.config/UltraGX (Linux y macOS)

Soporte

Escribinos a contact@ultra-gx.com con:

- la versión de UltraGX (**Configuración → Entorno → Aplicación**) y el sistema operativo;
- el mensaje de error o el final del log de la tarea;
- `run_summary.json` y `acceptance_report.md` del proyecto, si el problema es de una ejecución.

No mandes el XPZ, el código de la KB ni la carpeta de salida completa salvo que te lo pidamos y tu organización lo autorice.

Preguntas frecuentes

¿UltraGX modifica mi KB? No. Solo lee: el conector no tiene ningún camino de escritura y con un XPZ ni siquiera toca la KB.

¿Necesito GeneXus instalado? Solo para la fuente live. Con un XPZ o un export de texto, no.

¿Qué versiones de GeneXus soporta? De la 7.5 a la 18 por XPZ/XML; de X Evolution 1 a 18 en vivo.

¿Puedo documentar la misma KB muchas veces? Sí. Con el mismo nombre de proyecto se regenera en la misma carpeta y se conservan notas y narrativa. En la demo, la muestra de 25 objetos es siempre la misma.

¿Puedo compartir la documentación? Sí: la carpeta del proyecto es autocontenida y el visor funciona sin conexión. Tené en cuenta que contiene el código de la KB.

¿El código fuente se envía a algún lado? Solo si usás la narrativa con un proveedor en la nube, y a ese proveedor. Ver [Narrativa con IA](#).

Glosario

Término	Significado
Aceptación	Control de calidad que el motor ejecuta al final de cada pipeline: <code>passed</code> , <code>warning</code> o <code>failed</code> .
Anexo determinista	Última parte de cada documento de objeto con la evidencia técnica completa del análisis.
Árbol de llamadas	Diagrama de quién llama a quién a partir de un objeto o de un punto de entrada.
Cluster	Grupo de objetos que trabajan juntos en una misma área funcional, inferido del grafo de dependencias y de la ubicación en la KB.
Conector	Componente de UltraGX para Windows que abre una KB con el GeneXus instalado, en modo solo lectura, y la exporta a texto.
Confianza	Puntaje de 0 a 100 (<code>high</code> , <code>medium</code> , <code>low</code>) que indica cuánta evidencia respaldó el análisis de un objeto.
Conexión	Proveedor, modelo y credenciales de IA guardados con un nombre, para la narrativa.
Declaración	Objeto que solo define un tipo, una estructura o una configuración (atributo, dominio, SDT, objeto externo, configuración de patrón). No se narra por defecto.
degraded	Resultado de la narrativa de un objeto cuando el modelo citó nombres que no existen en él. No se publica.
Diagrama de flujo de datos (DFD)	Diagrama de qué objetos leen y escriben qué tablas.
Edición demo / completa	Tipos de licencia. La demo dura 10 días y detalla 25 objetos; la completa documenta todo sin conexión.
Export de texto	Formato de GeneXus 17 y 18 que guarda cada objeto como archivo de texto, en carpetas <code>src/</code> y <code>ref/</code> .
Hoja de ruta de reconstrucción	Orden sugerido para reconstruir la aplicación, por fases, según las dependencias entre clusters.
KB (Knowledge Base)	La base de conocimiento de GeneXus: todos los objetos de una aplicación.
Knowledge pack	Resumen estructurado de un objeto: comprensión funcional, modelo de interacción, uso de datos, dependencias, riesgos y lagunas.
Live (fuente)	Lectura de la KB directamente con el GeneXus instalado, a través del conector.
Motor	<code>ultragx-engine</code> , el programa que analiza la KB y escribe la documentación. La aplicación lo ejecuta por vos.
Narrativa	Prosa escrita por un modelo de lenguaje y validada contra los hechos del objeto.
Notas del usuario	Comentarios que el equipo agrega a los objetos; se conservan entre ejecuciones y prevalecen sobre lo inferido.
Objeto de librería	Objeto de un módulo importado (de GeneXus o de terceros), no desarrollado en la KB.
Objeto de negocio	Transacciones, Web Panels, Work Panels, Procedimientos, Reportes, Query Objects y SDTs: donde vive la lógica de la aplicación.

Término	Significado
Perfil funcional	Clasificación de lo que hace un objeto (por ejemplo, mantenimiento de datos, consulta, proceso por lotes).
Pipeline	La ejecución completa del motor sobre una KB: lectura, análisis, relaciones y escritura.
Proyecto	La carpeta de salida de una KB documentada.
Pseudocódigo	Traducción legible de cada evento y subrutina del objeto.
SQL implícito	La consulta que GeneXus genera para un <code>For each</code> , reconstruida por el motor: tabla base, joins, condiciones y orden.
Stub	En la demo, documento resumido de un objeto fuera de la muestra.
Visor	Sitio web estático generado para cada proyecto, navegable sin conexión.
XPZ	Archivo de exportación de objetos de GeneXus (XML comprimido).