



Manual do UltraGX

Instalação, licença e uso do UltraGX: do download a uma KB documentada e pronta para migrar.

Versão 0.2.8 · ultra-gx.com

Conteúdo

01 Introdução

02 Instalação

03 Primeiro início e licença

04 O aplicativo

05 Documentar uma KB

06 Ler o resultado

07 Narrativa com IA

08 Notas do usuário

09 Migrar com um agente de IA

10 Linha de comando

11 Solução de problemas e perguntas frequentes

12 Glossário

Introdução

O UltraGX lê uma Knowledge Base (KB) do GeneXus, da versão 7.5 à 18, e deixa tudo o que sabe sobre ela em arquivos abertos: documentos Markdown para ler, JSON para processar e um site navegável que funciona sem conexão. O resultado serve para três coisas:

- **Entender o sistema** sem abrir a IDE do GeneXus: o que cada objeto faz, quais regras de negócio aplica, quais tabelas lê e grava, quem o chama e quem ele chama.
- **Preservar o conhecimento** em um formato que pode ser versionado, pesquisado e compartilhado, mesmo que as pessoas que construíram o sistema já não estejam mais lá.
- **Migrar**: a saída foi pensada para que um agente de IA (Claude Code, Codex, Cursor, Antigravity) regenere o aplicativo em outra tecnologia, módulo por módulo, sem reinterpretar o GeneXus do zero.

O que ele faz, passo a passo

1. **Lê a KB** a partir de um export XPZ/XML, de um export de texto ou, no Windows com o GeneXus instalado, abrindo a KB diretamente com o conector.
2. **Analisa cada objeto** com regras determinísticas: tipo, estrutura, regras, eventos, variáveis, navegação, acessos a dados e referências. Nesta etapa não há palpites de um modelo de linguagem: cada conclusão fica ligada à evidência que a produziu.
3. **Relaciona os objetos**: monta o grafo de dependências, o modelo de dados, as árvores de chamadas, os diagramas de fluxo de dados e os *clusters* (grupos funcionais de objetos que trabalham juntos).
4. **Escreve a documentação**: um documento por objeto, o modelo de dados, os clusters, um roteiro de reconstrução e o visualizador web.
5. **Opcionalmente, adiciona prosa com IA (a narrativa)**: um modelo de linguagem redige explicações e o motor verifica se cada nome citado existe no objeto antes de publicá-la.

Somente leitura

O UltraGX **nunca modifica a KB**. O conector abre a Knowledge Base em modo de leitura e não tem nenhum caminho de escrita; com um XPZ ou um export de texto nem é preciso ter o GeneXus instalado. Tudo o que ele gera é gravado em uma pasta de saída separada.

Edições

	Demo	Completa
Duração	10 dias a partir da instalação	conforme a licença
Inventário, grafo, modelo de dados, clusters, árvores de chamadas e diagramas	toda a KB	toda a KB

	Demo	Completa
Documento detalhado por objeto (regras, pseudocódigo, SQL implícito, interface, código original)	25 objetos escolhidos pelo motor	todos
Knowledge packs, planos de reconstrução por cluster, narrativa com IA	somente para a amostra	todos
Conexão com a internet	necessária em cada execução (verificação da licença)	não é necessária

Na demo, os demais objetos recebem um documento resumido com sua identidade, localização, perfil, confiança, cluster e referências, e um aviso de que o detalhe está na edição completa.

Quais dados saem da sua máquina

- **A análise da KB é feita inteiramente no seu computador.** O código, as regras e a estrutura dos seus objetos não são enviados a nenhum servidor do UltraGX.
- **Licença demo:** ao se registrar, você envia seus dados de contato; em cada execução o serviço de licenças recebe o identificador da licença, o id da máquina, **quantos** objetos a KB tem (apenas quantidades) e os **nomes** dos objetos propostos para a amostra.
- **Licença completa:** não há comunicação com o serviço depois de ativá-la.
- **Narrativa com IA:** se você a usar, o documento de cada objeto é enviado ao provedor que você escolher (Anthropic, OpenAI, Google etc.) com a sua própria API key. Se preferir que nada saia da rede, use um modelo local (Ollama ou LM Studio). Veja [Narrativa com IA](#).

Como ler este manual

- **Se você está testando a demo:** [Instalação](#) → [Primeiro início e licença](#) → [Documentar uma KB](#) → [Ler o resultado](#).
- **Se você já tem licença:** [Documentar uma KB](#) → [Narrativa com IA](#) → [Notas do usuário](#).
- **Se você faz parte da equipe de migração:** [Ler o resultado](#) → [Migrar com um agente de IA](#) → [Linha de comando](#).

Instalação

Todos os downloads estão em ultra-gx.com. Ao lado de cada arquivo é publicado o seu hash SHA-256, caso você queira verificar se o download chegou íntegro.

Requisitos

Sistema	Versão mínima	O que mais é necessário
Windows	Windows 10 versão 1809 ou Windows 11, 64 bits	O runtime Microsoft Edge WebView2 (o instalador o adiciona se estiver faltando)
Linux	Debian 12, Ubuntu 22.04, Fedora 39 ou equivalentes, 64 bits	libwebkit2gtk-4.1 (ou 4.0) e GTK 3
macOS	macOS 13, Apple Silicon ou Intel	Nada

Não é preciso instalar Python, Node nem .NET: o motor e o aplicativo já vêm compilados.

Para ler uma KB ao vivo (sem exportá-la antes) é preciso Windows com o GeneXus X Evolution 1 a 18 instalado na mesma máquina. Para qualquer outro caso basta um export XPZ/XML ou um export de texto, e aí o sistema operacional não importa.

Espaço em disco: uma KB de 700 objetos gera cerca de 100 MB de documentação; uma de 5.000 objetos, em torno de 1 GB.

Windows

Com o instalador (recomendado)

1. Baixe `UltraGX-<versão>-win-x64-setup.exe`.
2. Execute-o. Se o Windows mostrar "**O Windows protegeu o computador**" (SmartScreen), clique em **Mais informações** e depois em **Executar assim mesmo**. O aviso aparece porque o instalador ainda é novo para o SmartScreen, não porque haja um problema com o arquivo.
3. **Idioma:** escolha English, Español ou Português. Esse idioma é usado no assistente, no contrato de licença exibido e como idioma inicial do aplicativo e dos documentos que ele gera. Os dois podem ser alterados depois.
4. **Contrato de licença:** leia e aceite para continuar.
5. **Pasta de destino:** por padrão `C:\Users\<usuário>\AppData\Local\Programs\UltraGX`. A instalação é por usuário e não pede permissões de administrador.
6. **Tarefas adicionais** (ambas opcionais):
 - *Criar um ícone na Área de Trabalho.*
 - *Abrir exports do GeneXus (.xpz) com o UltraGX:* com esta opção, ao dar um clique duplo em um `.xpz`, o UltraGX abre com a aba **Executar pipeline** pronta para documentá-lo.
7. **Instalar.** Se faltar o WebView2 no computador, o instalador o baixa e instala nesta etapa (é preciso conexão com a internet).

8. Ao terminar, deixe marcada a opção **Executar UltraGX** para abri-lo e continue em [Primeiro início e licença](#).

No Windows, o instalador também inclui o **conector do GeneXus**, que permite ler uma KB ao vivo. Se não houver GeneXus instalado na máquina, o conector fica inativo e não atrapalha.

Linux

Escolha o formato adequado à sua distribuição:

```
# Debian / Ubuntu
sudo apt install ./ultragx-<versão>_amd64.deb

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

Os pacotes instalam o aplicativo em `/opt/ultragx`, o comando `ultragx` e a entrada no menu de aplicativos, e declaram as dependências do WebKitGTK, então o gerenciador de pacotes as instala automaticamente.

AppImage (sem instalar): dê permissão de execução e abra-o. Nesse caso as dependências precisam estar presentes no sistema:

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

No Linux a fonte "KB ao vivo" não está disponível (o conector precisa do GeneXus, que é para Windows): exporte a KB para XPZ ou para texto na máquina onde está o GeneXus e documente-a aqui.

macOS

1. Baixe o `.dmg` adequado ao seu Mac: `osx-arm64` para Apple Silicon (M1 em diante), `osx-x64` para Intel.
2. Abra-o e arraste o **UltraGX** para **Aplicativos**.
3. A versão para macOS não é assinada com um Developer ID da Apple, por isso o macOS a bloqueia na primeira vez. Dê dois cliques em **Allow UltraGX.command** na imagem de disco (ou execute `xattr -cr /Applications/UltraGX.app` no Terminal) e abra o UltraGX. Se o macOS insistir, vá em **Ajustes do Sistema → Privacidade e Segurança** e clique em **Abrir Mesmo Assim** ao lado da mensagem do UltraGX.

Assim como no Linux, no macOS se trabalha com exports XPZ ou de texto.

Atualizar

Baixe a nova versão e instale-a por cima da anterior (no Windows, o instalador substitui os arquivos na mesma pasta). As configurações, as conexões de narrativa, a licença e as saídas geradas são

preservadas.

As saídas geradas com uma versão anterior continuam podendo ser abertas. Para aproveitar as melhorias do novo motor, execute novamente o pipeline sobre a mesma KB: as notas do usuário e a narrativa já gerada são mantidas.

Desinstalar

- **Windows:** *Configurações → Aplicativos → Aplicativos instalados → UltraGX → Desinstalar.*
- **Linux:** `sudo apt remove ultragx` ou `sudo dnf remove ultragx` ; a Applmage basta apagar.
- **macOS:** arraste o UltraGX de Aplicativos para o Lixo.

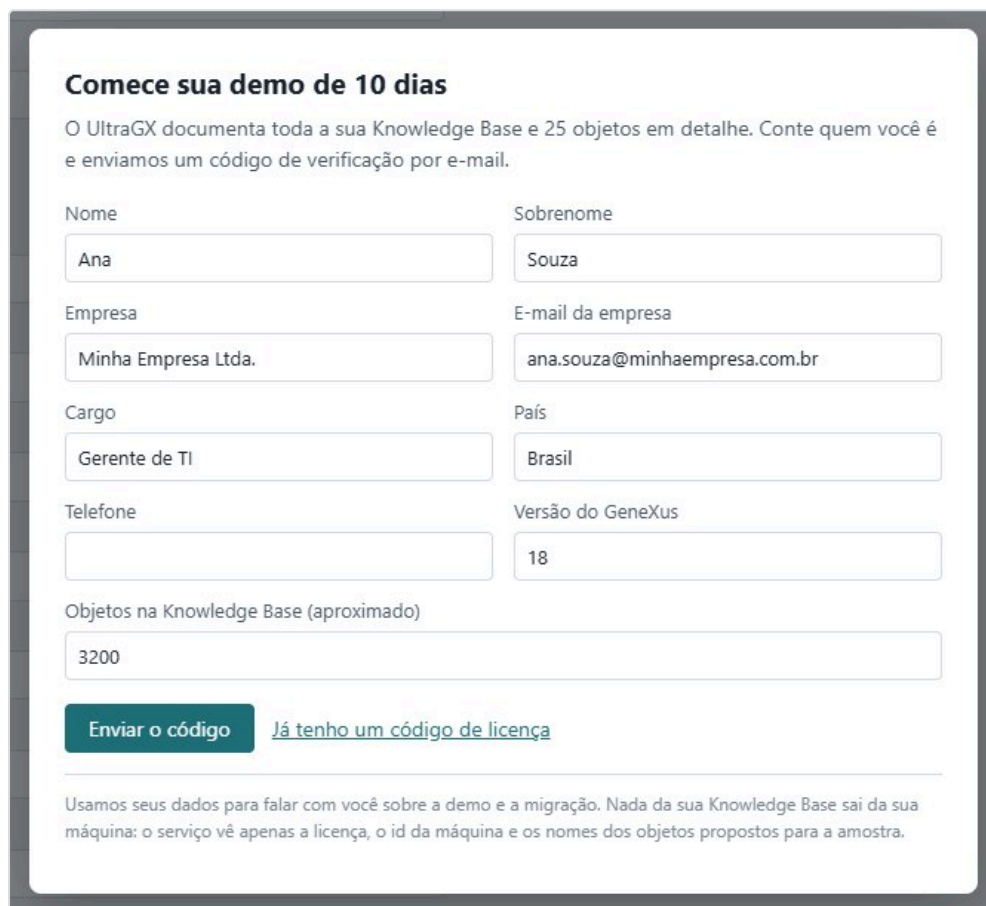
Desinstalar **não apaga** os seus dados:

O quê	Onde
Configurações, conexões de narrativa (incluindo as API keys) e licença	%APPDATA%\UltraGX no Windows, ~/.config/UltraGX no Linux e no macOS
Saídas geradas	a pasta de saída que você escolheu
Registro do período de demo	é mantido de propósito: reinstalar não reinicia os 10 dias

Se quiser remover tudo, apague também a pasta de configurações e a sua pasta de saída. Lembre-se de que as configurações guardam as suas API keys.

Primeiro início e licença

O UltraGX precisa de uma licença para executar o motor. Na primeira vez que você abre o aplicativo sem licença, aparece o formulário **Comece sua demo de 10 dias**. A partir dele você pode pedir a demo ou, se já comprou a edição completa, usar o seu código.



The screenshot shows a web form titled "Comece sua demo de 10 dias". Below the title is a paragraph: "O UltraGX documenta toda a sua Knowledge Base e 25 objetos em detalhe. Conte quem você é e enviamos um código de verificação por e-mail." The form contains several input fields arranged in two columns. The left column has fields for "Nome" (containing "Ana"), "Empresa" (containing "Minha Empresa Ltda."), "Cargo" (containing "Gerente de TI"), "Telefone" (empty), and "Objetos na Knowledge Base (aproximado)" (containing "3200"). The right column has fields for "Sobrenome" (containing "Souza"), "E-mail da empresa" (containing "ana.souza@minhaempresa.com.br"), "País" (containing "Brasil"), and "Versão do GeneXus" (containing "18"). At the bottom left is a green button labeled "Enviar o código". To its right is a link labeled "Já tenho um código de licença". Below the form, there is a small disclaimer: "Usamos seus dados para falar com você sobre a demo e a migração. Nada da sua Knowledge Base sai da sua máquina: o serviço vê apenas a licença, o id da máquina e os nomes dos objetos propostos para a amostra."

Formulário de registro da demo

Pedir a demo

1. Preencha o formulário. São obrigatórios **Nome, Sobrenome, Empresa e E-mail da empresa**; o restante (**Cargo, País, Telefone, Versão do GeneXus, Objetos na Knowledge Base**) nos ajuda a preparar a conversa sobre a migração.
2. Use o **e-mail da sua empresa**. A demo é entregue a empresas que estão avaliando uma migração e o serviço recusa e-mails pessoais (Gmail, Outlook, Hotmail, Yahoo, iCloud e similares).
3. Clique em **Enviar o código**. Em alguns minutos chega um e-mail com um código de seis dígitos.
4. Digite o código e clique em **Ativar a demo**. O código vale por **15 minutos**.

Comece sua demo de 10 dias

O UltraGX documenta toda a sua Knowledge Base e 25 objetos em detalhe. Conte quem você é e enviamos um código de verificação por e-mail.

Enviamos um código de seis dígitos para ana.souza@minhaempresa.com.br. Ele vale por 15 minutos.

4 8 2 9 1 3

Ativar a demo

[Voltar](#)

Não chegou? Veja a pasta de spam, ou volte e tente outro endereço.

Usamos seus dados para falar com você sobre a demo e a migração. Nada da sua Knowledge Base sai da sua máquina: o serviço vê apenas a licença, o id da máquina e os nomes dos objetos propostos para a amostra.

Etapa do código de verificação

Se o e-mail não chegar, veja a pasta de spam. Se ele continuar sem aparecer, clique em **Voltar** e peça o código de novo (ou tente outro endereço da empresa).

Quando a demo fica ativa, o formulário se fecha e o aplicativo mostra o aviso **Edição DEMO** na barra superior.

O que a demo inclui

- **10 dias** contados a partir da instalação naquela máquina. Reinstalar ou pedir outro código não reinicia o prazo.
- Uma demo por máquina e uma por e-mail.
- A KB completa é analisada e documentada no nível de inventário, modelo de dados, grafo, clusters, árvores de chamadas e diagramas. **25 objetos** recebem o documento detalhado; o motor os escolhe entre os objetos de negócio mais conectados, distribuídos por tipo, para que a amostra seja representativa. Os mesmos 25 se mantêm mesmo que você execute a KB várias vezes.
- Cada execução **precisa de internet** para consultar o serviço de licenças. Se não houver conexão, a execução termina mesmo assim, mas sem objetos detalhados, e o aviso fica no README e no visualizador.

Ativar a edição completa

Quando você comprar a licença, vai receber um código no formato ULGX-XXXX....

- **No primeiro início:** no formulário, clique em **Já tenho um código de licença**, cole o código e clique em **Ativar com um código**.
- **Com o aplicativo já em uso** (por exemplo, ao passar da demo para a completa): vá em **Configurações → Licença**, cole o código em **Código de ativação** e clique em **Ativar com um**

código. O campo **URL do serviço de licenças** deve ficar vazio, a menos que indiquemos outra coisa.

A ativação vincula a licença àquela máquina. Depois de ativada, a edição completa não volta a se conectar ao serviço.

Instalar um arquivo de licença

Em instalações sem acesso à internet, podemos entregar um arquivo `ultragx.lic` em vez de um código:

1. Em **Configurações → Licença**, copie o **Id desta máquina** e envie-o para nós.
2. Quando receber o arquivo, clique em **Instalar arquivo de licença...** e selecione-o.

O arquivo fica em `%APPDATA%\UltraGX\ultragx.lic` (Windows) ou `~/.config/UltraGX/ultragx.lic` (Linux e macOS).

Licença

Status	válida
Edição	completa
Titular	Minha Empresa Ltda. <sistemas@minhaempresa.com.br>
Expira	2036-09-04
Arquivo de licença	%APPDATA%\UltraGX\ultragx.lic
Id desta máquina	3f9c2a71b8e4d05c (envie ao proprietário para obter uma licença vinculada a esta máquina)

Código de ativação (DEMO-XXXX / ULGX)

URL do serviço de licenças (opcional)

Ativar com um código

Instalar arquivo de licença...

Configurações, bloco Licença

Ver o status da licença

Configurações → Licença mostra:

Campo	O que significa
Status	válida, ou o motivo pelo qual não é
Edição	DEMO com a quantidade de objetos detalhados e o serviço que consulta, ou completa
Titular	empresa e e-mail para os quais foi emitida
Expira	data de vencimento, ou nunca
Arquivo de licença	onde está instalada
Id desta máquina	identificador usado para vincular a licença

Mensagens do registro e da ativação

Mensagem	Causa	O que fazer
Use o e-mail da sua empresa...	Você informou um e-mail pessoal	Registre-se com o endereço da empresa

Mensagem	Causa	O que fazer
Este e-mail já ativou uma demo...	Esse e-mail já foi usado	Escreva para contact@ultra-gx.com se precisar de mais tempo
O período de demo nesta máquina terminou...	Os 10 dias se passaram	Escreva para nós para seguir com a edição completa
Código incorreto...	O código não confere	Confira o último e-mail recebido e digite-o de novo
O código expirou...	Passaram-se mais de 15 minutos	Voltar e pedir um novo
Códigos incorretos demais...	Cinco tentativas com erro	Voltar e começar de novo
Tentativas demais nesta rede...	Muitas solicitações seguidas a partir da mesma rede	Espere um pouco e tente de novo
Confira os campos...	Falta um campo obrigatório ou ele não é válido	Preencha nome, sobrenome, empresa e um e-mail válido
Não foi possível contatar o serviço de licenças...	Sem internet, ou um proxy/firewall bloqueia licenses.ultra-gx.com	Verifique a conexão; em redes corporativas, peça que liberem HTTPS para licenses.ultra-gx.com

O aplicativo

A janela do UltraGX tem uma barra superior com seis abas e, à direita, a **pasta de saída** ativa com o botão **Pasta de saída...** para alterá-la. Cada execução do pipeline cria uma subpasta dentro dessa pasta; convém escolher uma em um disco com espaço e fora de pastas sincronizadas (OneDrive, Dropbox), porque uma KB grande gera milhares de arquivos.

Na primeira vez, se não houver pasta de saída definida, o aplicativo abre diretamente **Configurações**.

Projetos

Projeto	Fonte	Entrada	Objetos	Clusters	Conf. média	Aceitação	Narrativa ok	Modificado
FacturasDemo	xpz	It\models\UltraGX\dist\manual-input\FacturasDemo.xml	696	324	82.0	passed		2026-09-16 18:24

Aba Projetos

Lista cada pasta sob a pasta de saída que contém uma execução do pipeline:

Coluna	O que mostra
Projeto	nome da subpasta
Fonte / Entrada	de onde a KB foi lida (xpz , gxttext , live) e o caminho
Objetos / Clusters	quantidade de objetos documentados e de grupos funcionais
Conf. média	confiança média da análise, de 0 a 100
Aceitação	resultado do controle de qualidade final (passed , warning , failed)
Narrativa ok	objetos com prosa de IA publicada
Modificado	data da última execução

Abrir leva o projeto para o **Visualizador**; **Regerar visualizador** reconstrói o site; **Pasta** o abre no gerenciador de arquivos. **Atualizar** lê de novo a pasta de saída e **Abrir pasta** abre a pasta de saída inteira.

Visualizador

Mostra o site gerado para o projeto escolhido na lista suspensa. Ele é explicado em detalhe em [Ler o resultado](#).

- **Recarregar**: recarrega a página (útil enquanto a narrativa está rodando, pois ela vai atualizando os documentos).
- **Regerar visualizador**: reconstrói o site a partir dos arquivos da saída, sem analisar a KB de novo.
- **Abrir no navegador**: abre o mesmo visualizador no seu navegador. Pelo navegador é possível ler tudo, mas as notas do usuário só podem ser adicionadas pelo aplicativo.

Executar pipeline

É onde uma KB é documentada: escolhe-se a fonte, a entrada, o nome do projeto e o idioma dos documentos. Veja [Documentar uma KB](#).

Narrativa

Adiciona prosa escrita por um modelo de linguagem aos documentos de um projeto já gerado, com conexões a diferentes provedores. Veja [Narrativa com IA](#).

Tarefas

UltraGX

Projetos

Visualizador

Executar pipeline

Narrativa

Tarefas

Configurações

lt\models\UltraGX\dist\manual-shots-pt-BR

Pasta de saída...

Tarefas

Tipo	Projeto	Status	Início	Comando	
pipeline	FacturasDemo	ok	2026-09-16 18:24:21	%LOCALAPPDATA%\Programs\UltraGX\ultrax-engine.exe pipeline --source xpt --out I:\models\UltraGX\dist\manual-shots-pt-BR\FacturasDemo --lang pt-BR	Log
pipeline	FacturasDemo	ok	2026-09-16 18:17:00	%LOCALAPPDATA%\Programs\UltraGX\ultrax-engine.exe pipeline --source xpt --out I:\models\UltraGX\dist\manual-shots-en\FacturasDemo --lang en --inp	Log

Aba Tarefas

Cada pipeline e cada narrativa que você inicia vira uma tarefa com seu **Tipo**, **Status**, **Início** e **Comando**. **Log** mostra a saída completa da tarefa (também das que já terminaram, enquanto o aplicativo continuar aberto) e **Parar** a interrompe. É possível executar um pipeline e uma narrativa ao mesmo tempo, mas não dois pipelines nem duas narrativas simultâneas.

Não feche o aplicativo com tarefas em andamento: o acompanhamento e o log se perdem. Espere que terminem ou pare-as antes. Uma narrativa interrompida pode ser **retomada** depois sem repetir os objetos já processados.

Configurações

UltraGX

Projetos

Visualizador

Executar pipeline

Narrativa

Tarefas

Configurações

Configurações

Idioma da interface

Português (BR)

Pasta de saída (uma subpasta por execução)

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

Procurar...

Salvar

salvo

Licença

Status	válida
Edição	completa
Titular	Minha Empresa Ltda. <sistemas@minhaempresa.com.br>
Expira	2036-09-04
Arquivo de licença	%APPDATA%\UltraGX\ultragx.lic
Id desta máquina	3f9c2a71b8e4d05c (envie ao proprietário para obter uma licença vinculada a esta máquina)

Código de ativação (DEMO-XXXX / ULGX)

URL do serviço de licenças (opcional)

Ativar com um código

Instalar arquivo de licença...

Ambiente

Aplicativo	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)
Pasta de saída	ok
Arquivo de configurações	%APPDATA%\UltraGX\settings.json (or ~/.config/UltraGX)

Aba Configurações

- **Idioma da interface:** English, Español ou Português (BR). Não altera o idioma dos documentos, que é escolhido em cada execução.
- **Pasta de saída:** a mesma que a barra superior mostra. Clique em **Salvar** depois de alterá-la.
- **Licença:** status, ativação com código e instalação de um arquivo de licença. Veja [Primeiro início e licença](#).
- **Ambiente:** o que o aplicativo detectou na máquina.

Ambiente	O que significa
Aplicativo	versão e sistema operacional
Motor	caminho e versão do motor compilado em uso
Conector GeneXus	caminho do conector, ou não encontrado (no Linux e no macOS sempre; no Windows só se for usada uma cópia sem conector)
GeneXus instalado	instalações do GeneXus encontradas, com sua versão; se não houver nenhuma, trabalha-se com XPZ ou export de texto
Pasta de saída	ok se existir
Arquivo de configurações	onde ficam guardadas as configurações, as conexões e a licença

Documentar uma KB

Documentar uma Knowledge Base é executar o **pipeline** sobre ela: a aba **Executar pipeline** lê a KB, a analisa e grava a documentação completa em uma subpasta da pasta de saída.

UltraGX Projetos Visualizador **Executar pipeline** Narrativa Tarefas Configurações

Executar o pipeline

Fonte

xpz — export XPZ / XML (GeneXus 7.5 ao 18) ▼

Lê exports XPZ ou XML do GeneXus 7.5, 8 e 9 (o formato antigo KMW 2.x) e do X Evolution 1 ao 18. A versão é detectada sozinha, não há nada para escolher.

Entrada

I:\models\UltraGX\dist>manual-input\FacturasDemo.xml Procurar...

Nome do projeto (subpasta de saída)

FacturasDemo

Idioma dos documentos

Português (BR) ▼

☐ gxttext: ignorar módulos referenciados (ref/)

Executar

▶ Entradas recentes

Aba Executar pipeline

1. Escolher a fonte

Fonte	O que se escolhe em Entrada	Quando usar
xpz — export XPZ / XML (GeneXus 7.5 ao 18)	um arquivo <code>.xpz</code> ou <code>.xml</code>	O caso geral. Serve para qualquer versão e em qualquer sistema operacional. A versão do GeneXus é detectada sozinha
gxttext — pasta de export de texto (src/ + ref/)	a pasta que contém <code>src/</code> e <code>ref/</code>	Se você já tem um export de texto do GeneXus 17 ou 18
live — abrir a KB com o GeneXus instalado nesta máquina (conector)	a pasta da KB (a que contém o <code>.gxw</code>)	Windows com GeneXus X Evolution 1 a 18 instalado. Não é preciso exportar nada manualmente
mcp — pasta de dumps JSON	uma pasta de dumps JSON	Uso interno e suporte; normalmente não é usada

Abaixo do seletor, o aplicativo explica cada fonte. Se a opção **live** aparecer desabilitada (*requer o conector e uma instalação do GeneXus*), a máquina não tem GeneXus ou está sendo usada uma cópia sem conector.

Como obter um XPZ

Na IDE do GeneXus: **Knowledge Manager → Export**, marque todos os objetos da KB e exporte para um arquivo `.xpz`. Em versões antigas (7.5, 8, 9) o caminho equivalente é **Knowledge Manager → Distribute**. Inclua tudo: quanto mais completo o export, melhores ficam as referências, o modelo de dados e os clusters.

Dicas:

- Exporte a partir da **KB de desenvolvimento** atualizada, não de uma cópia antiga.
- Se a KB for muito grande e a IDE não conseguir exportá-la de uma vez, exporte por partes (por módulo ou por tipo de objeto) e documente cada arquivo como um projeto separado, ou use a fonte **live**.
- Se você documentou a KB com uma versão do UltraGX anterior à 0.2.5 e os SDTs aparecem sem membros, execute novamente o pipeline (e, com a fonte live, a exportação) com a versão atual.

Fonte live e versão do GeneXus

Com **live**, o UltraGX abre a KB com o GeneXus instalado, a exporta para texto dentro da pasta do projeto (`_source_text/`) e depois executa o pipeline. Em **Versão do GeneXus para a KB ao vivo**, deixe **auto (a partir da KB)**: o conector escolhe a instalação que corresponde à versão com que a KB foi salva. Se você tiver várias instalações da mesma versão (por exemplo, dois *upgrades* do GeneXus 18), pode forçar uma delas.

Antes de executar com **live**:

- **Feche a KB na IDE do GeneXus.** Uma KB só pode estar aberta em um processo por vez; se a IDE (ou outra ferramenta) a tiver aberta, a exportação espera ou falha.
- Conte com o fato de que a exportação ao vivo é a etapa mais lenta: o conector extrai cada objeto pelo SDK do GeneXus. Uma KB de 5.000 objetos pode levar mais de uma hora só para exportar. O log vai mostrando qual objeto está extraíndo.

2. Nome do projeto e idioma

- **Nome do projeto (subpasta de saída):** por padrão, o nome do arquivo ou da pasta de entrada. Se você executar de novo com o mesmo nome, o projeto é regenerado na mesma pasta e as **notas do usuário** e a **narrativa** já gerada são preservadas.
- **Idioma dos documentos:** English, Español ou Português (BR). É o idioma da prosa dos documentos e do visualizador. O `README.md` da raiz da saída fica sempre em inglês, porque foi pensado para agentes de IA.
- **gxttext: ignorar módulos referenciados (ref/):** com as fontes gxttext e live, deixa de fora os objetos de módulos que a KB apenas referencia (bibliotecas do GeneXus e de terceiros). Acelera a execução e limpa o inventário; a contrapartida é que as chamadas a esses módulos ficam como referências externas.

3. Executar

Clique em **Executar**. O log abaixo mostra o avanço por etapas:

```
[ 2.7s] analysing 578/696 WebPanel.GAMChangeRepository
[ 3.7s] analysing 601/696 WebPanel.GAMUserPermissionSelect
[ 4.7s] analysing 696/696 WorkWithPlusInstance.WorkWithPlusFactura
[ 4.7s] extracted 696 objects
[ 4.7s] semantic rules and validation on 696 objects
[ 5.6s] indexing the KB
[ 5.6s] resolving references and building the graph
[ 5.8s] clustering
[ 5.9s] rebuild plan per cluster
[ 6.0s] cluster knowledge and functional flow
[ 6.6s] writing 696 object documents
[ 6.6s] data model (foreign keys and ERDs)
[ 6.6s] writing 1/696 Procedure.GoogleDirectionsServiceRequest
[ 7.6s] writing 254/696 Domain.GAMSessionType
[ 8.6s] writing 464/696 Procedure.GetFilterByName
[ 9.6s] writing 595/696 WebPanel.GAMMMRoleRoles
[ 10.4s] writing 696/696 WorkWithPlusInstance.WorkWithPlusFactura
```

Log de uma execução

- **Leitura:** o motor lê a fonte e informa o objeto que está processando.
- **Análise:** regras, eventos, variáveis, acessos a dados e referências de cada objeto.
- **Grafo, clusters, conhecimento e plano de reconstrução:** relações entre objetos.
- **Escrita:** documentos, modelo de dados, diagramas e visualizador.
- **Aceitação:** controle de qualidade final.

Para não inundar a tela, o log escreve no máximo uma linha por segundo por etapa. Se uma linha se repetir com o mesmo objeto por algum tempo, o motor continua trabalhando em um objeto grande: ele não travou.

Quando termina bem, o status passa para **concluído**, o projeto aparece em **Projetos** e abre sozinho no **Visualizador**. Se a execução falhar, o status diz **terminou com status erro** e o log mostra o motivo (veja [Solução de problemas](#)).

Tempos de referência

KB	Fonte	Tempo aproximado
700 objetos	XPZ	menos de um minuto
5.000 objetos	XPZ ou export de texto	de alguns minutos a meia hora
5.000 objetos	live	a exportação (uma hora ou mais) mais o pipeline

Os tempos dependem do disco e do tamanho do código de cada objeto. Na demo, soma-se uma consulta breve ao serviço de licenças.

Entradas recentes

Entradas recentes guarda as últimas quinze entradas usadas. Clique em uma para carregá-la de novo em **Entrada**.

Abrir um XPZ com clique duplo

Se ao instalar você marcou *Abrir exports do GeneXus (.xpz) com o UltraGX*, o clique duplo em um **.xpz** abre o aplicativo em **Executar pipeline** com a fonte **xpz** e o arquivo já carregado: só falta revisar o nome do projeto e o idioma e clicar em **Executar**.

Ler o resultado

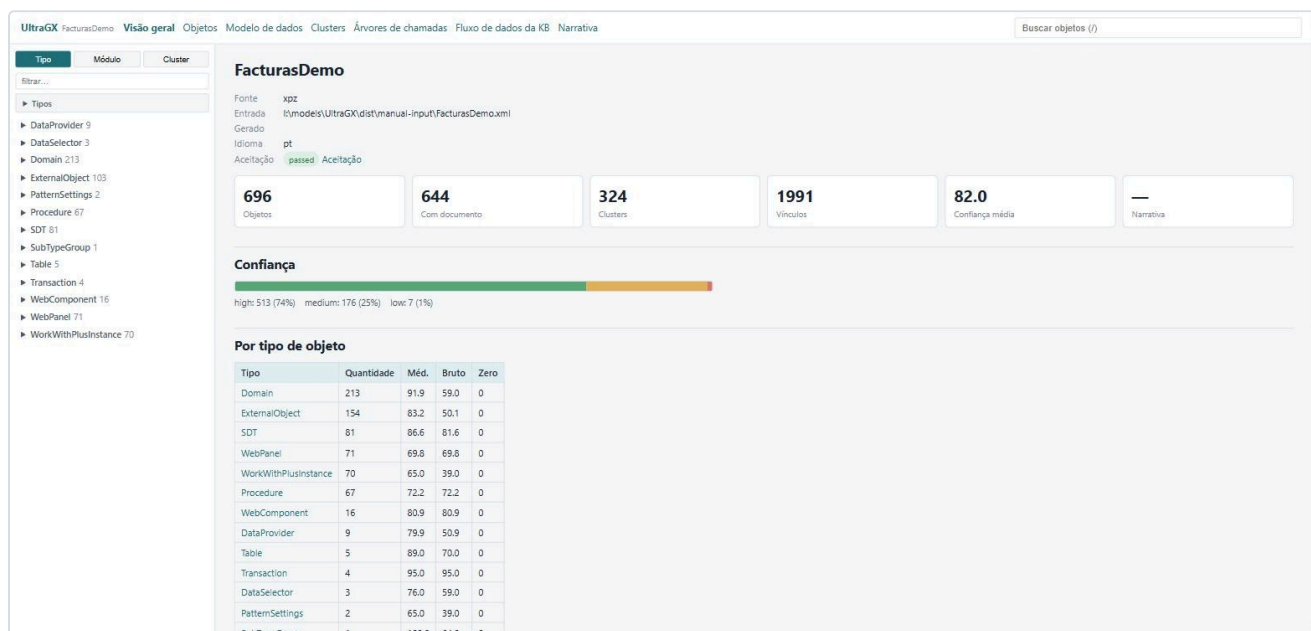
Cada execução deixa duas coisas: um **visualizador web** para ler e navegar, e uma **pasta de arquivos** Markdown e JSON para processar, versionar ou entregar a um agente de IA. Os dois têm o mesmo conteúdo.

O visualizador

O visualizador abre na aba **Visualizador** do aplicativo ou com **Abrir no navegador**. É um site estático que fica em `viewer/` dentro do projeto: funciona sem conexão e pode ser copiado para outra máquina junto com a pasta.

A barra superior tem as seções principais e a busca (**Buscar objetos**, atalho de teclado `/`), que encontra objetos pelo nome a partir de qualquer página.

Visão geral

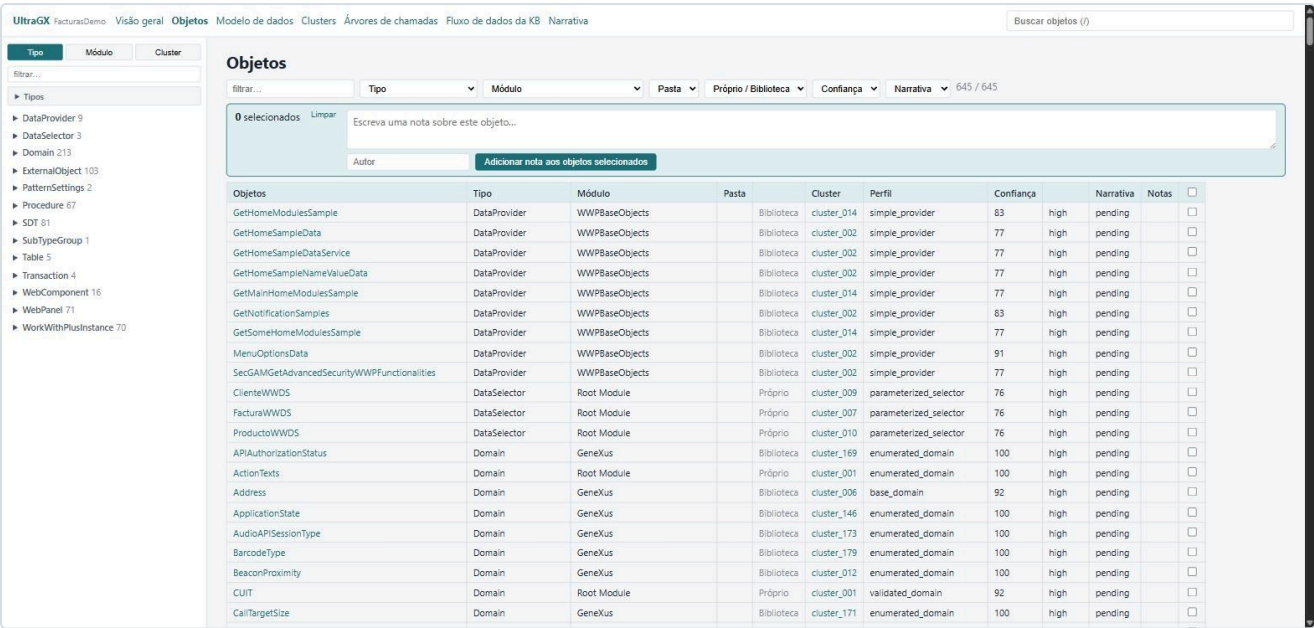


Visão geral do projeto

A página inicial mostra de onde veio a KB (fonte, entrada, data, idioma), o resultado do controle de qualidade e os números gerais: objetos, objetos com documento, clusters, relações, confiança média e status da narrativa. Abaixo estão a distribuição da **confiança**, a tabela por tipo de objeto, os maiores clusters e os links para os relatórios da KB (resumo, roteiro de reconstrução, validação, aceitação).

Confiança (0 a 100) mede quanta evidência o motor encontrou para descrever cada objeto: **high**, **medium** ou **low**. Uma confiança baixa não significa que o objeto esteja errado, e sim que ele tem pouca estrutura para analisar (por exemplo, um procedimento vazio ou um objeto de configuração).

Objetos

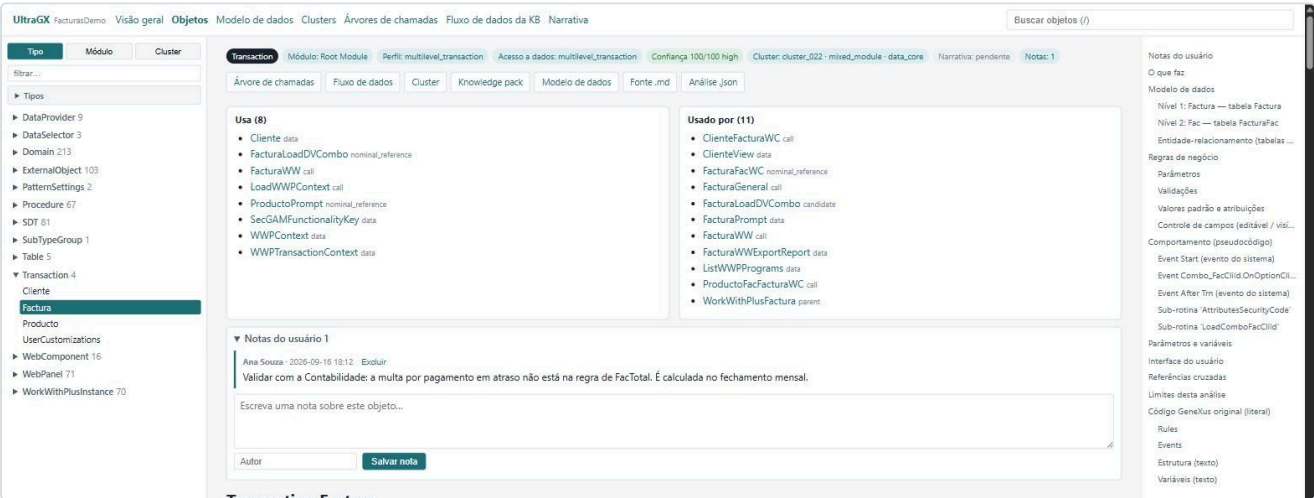


Lista de objetos

A lista de todos os objetos com filtros por texto, **Tipo** (incluindo a opção **Objetos de negócio**, que mantém Transações, Web Panels, Work Panels, Procedimentos, Relatórios, Query Objects e SDTs), **Módulo**, **Pasta**, **Próprio / Biblioteca**, **Confiança** e status da **Narrativa**. Acima da lista há uma caixa para adicionar uma nota aos objetos selecionados (veja **Notas do usuário**).

A **barra lateral**, presente em todas as páginas, agrupa os objetos por **Tipo**, **Módulo**, **Localização** (a mesma árvore de módulos e pastas do *KB Explorer* do GeneXus, quando a KB tem pastas) ou **Cluster**. Em **Tipos** se escolhe quais tipos mostrar e, se a KB tiver objetos de módulos importados, **Ocultar objetos de biblioteca** os deixa de fora.

Página de um objeto



Documento de um objeto

No topo estão a identidade do objeto (tipo, módulo, pasta, cluster, perfil funcional, acesso a dados, confiança), suas relações (**Usa** e **Usado por**, com links) e as **Notas do usuário**. Depois vem o

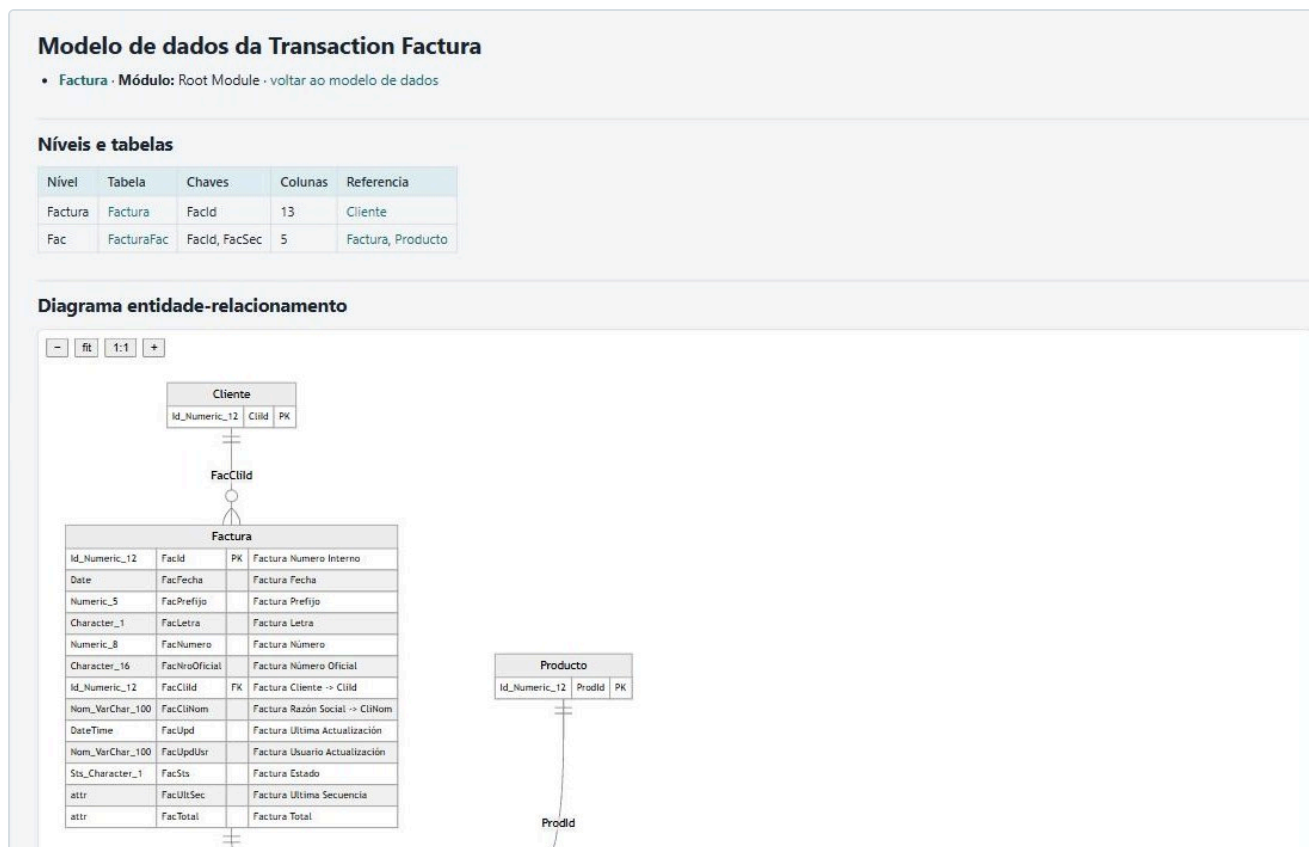
documento, com o índice de seções à direita:

Seção	O que contém
Resumo	tabela com os dados-chave do objeto
O que faz	descrição em prosa do seu propósito
Modelo de dados	(transações) níveis, atributos com seu papel (chave, chave estrangeira, fórmula, inferido) e diagrama entidade-relacionamento
Regras de negócio	cada regra, explicada uma a uma
Comportamento (pseudocódigo)	o que cada evento e sub-rotina faz, em pseudocódigo legível
SQL implícito (por For each)	a consulta que o GeneXus gera em cada For each : tabela base, joins, condições e ordem
Parâmetros e variáveis	a regra parm e as variáveis com seu tipo
Interface do usuário	campos, grids e sua tabela base, botões e ações, comportamento dinâmico
Significado de negócio	prosa da narrativa com IA, se foi gerada
Referências cruzadas	objetos relacionados
Limites desta análise	o que o motor não conseguiu determinar com certeza
Código GeneXus original (literal)	o código-fonte tal como está na KB

No final, recolhível, está o **Anexo determinístico do motor**: a evidência técnica completa da análise (achados, riscos, pontuação, rastreabilidade). Serve para auditar uma conclusão; para ler o objeto não é necessário.

As seções mudam conforme o tipo: um Data Provider mostra sua saída, um SDT sua estrutura, um Domain seus valores etc.

Modelo de dados



Modelo de dados

O diagrama entidade-relacionamento de toda a KB, a lista de tabelas e o detalhe por transação. Cada tabela tem sua página com colunas (chave primária e estrangeiras), relacionamentos, transações que a definem, objetos que a usam e um DDL conceitual.

Clusters

Um **cluster** é um grupo de objetos que trabalham juntos (por exemplo, tudo o que se relaciona com faturamento). O motor os monta a partir do grafo de dependências e da localização na KB. Cada cluster tem seu tipo, prioridade, objeto núcleo, pontos de entrada e membros, e três relatórios: **Conhecimento** (propósito, dados, riscos), **Fluxo funcional** (como o trabalho passa de um objeto a outro) e **Plano de reconstrução**.

O **Roteiro de reconstrução** (a partir da Visão geral) ordena os clusters em fases conforme suas dependências: primeiro o que não depende de nada, depois o que se apoia nisso.

Árvores de chamadas e fluxo de dados

- **Árvores de chamadas**: a partir de cada ponto de entrada, quem chama quem. Os nós do diagrama são clicáveis.
- **Fluxo de dados da KB**: quais objetos leem e gravam quais tabelas, para toda a KB, por cluster e por objeto.

Diagramas muito grandes não são desenhados automaticamente: aparece o botão **Desenhar diagrama**, e também é possível ver o texto com **Fonte do diagrama**.

Narrativa

Status da narrativa com IA por objeto: `ok`, `degraded`, `error` ou pendente. Veja [Narrativa com IA](#).

A pasta de saída

Tudo o que o visualizador mostra está em arquivos dentro da pasta do projeto:

Caminho	Conteúdo
<code>README.md</code>	Ponto de entrada: como usar a pasta, mapa, glossário, clusters, tabelas e inventário completo. Sempre em inglês
<code>run_summary.json</code>	Como foi gerada: fonte, entrada, idioma, quantidades, licença, aceitação
<code>kb_index.json</code>	Uma linha por objeto: tipo, módulo, pasta, perfil, confiança, cluster e <code>doc_path</code> , o caminho exato do seu documento
<code>kb_graph.json</code>	Grafo de dependências: relações de entrada e de saída de cada objeto
<code>kb_summary.md</code> / <code>.json</code>	Métricas da KB
<code>kb_clusters.json</code> , <code>kb_clusters_summary.md</code>	Clusters
<code>global_rebuild_roadmap.md</code> / <code>.json</code>	Roteiro de reconstrução por fases
<code>acceptance_report.md</code> / <code>.json</code>	Controle de qualidade final
<code>objects/</code>	Um <code><Tipo>.<Nome>.md</code> e um <code>.json</code> por objeto, em pastas que replicam os módulos e as pastas da KB
<code>knowledge/</code>	Knowledge pack de cada objeto
<code>datamodel/</code>	Modelo de dados: índice, uma página por tabela e por transação, <code>kb_datamodel.json</code>
<code>clusters/</code>	Conhecimento, fluxo e plano de reconstrução de cada cluster
<code>calltree/</code> , <code>dfd/</code>	Árvores de chamadas e diagramas de fluxo de dados
<code>narrative/</code>	Resultado da narrativa com IA, um JSON por objeto
<code>annotations/</code>	Notas do usuário
<code>viewer/</code>	O visualizador. É regenerado: não o edite manualmente
<code>_pipeline/</code> , <code>_source_text/</code>	Uso interno: snapshots para detectar mudanças e, com a fonte live, o export de texto

Os documentos são Markdown padrão com diagramas Mermaid: são bem lidos no GitHub, GitLab, VS Code ou Obsidian.

Narrativa com IA

O pipeline documenta a KB com regras determinísticas. A **narrativa** é uma etapa opcional que adiciona, a cada documento, prosa escrita por um modelo de linguagem: o propósito do objeto, um resumo, a intenção de cada evento, as regras explicadas em termos de negócio, notas para a migração e um glossário.

O que a diferencia de colar o código em um chat é a **validação**: o modelo recebe o documento já analisado (não o GeneXus bruto) e o UltraGX verifica se cada atributo, tabela, variável ou objeto citado existe realmente nesse objeto. Se o modelo inventar um nome, ele é solicitado a corrigir; se insistir, a prosa não é publicada.

A narrativa é executada sobre um projeto **já gerado** pelo pipeline.

The screenshot shows the 'Narrativa' (Narrative) configuration page in the UltraGX application. The page title is 'Narrativa (prosa de LLM validada contra os fatos)'. It contains several configuration fields:

- Projeto:** A dropdown menu with 'FacturasDemo' selected.
- Conexão:** A dropdown menu with 'Gemini (empresa)' selected. Below it are buttons for '+ Nova' and 'Excluir', and a text input field containing 'Gemini (empresa)'.
- Provedor:** A dropdown menu with 'Google (Gemini)' selected.
- Modelo:** A text input field with 'gemini-3.8-flash'.
- URL base (compatível com OpenAI):** A text input field with 'https://generativelanguage.googleapis.com/v1beta/openai/'.
- API key desta conexão (guardada no arquivo de configurações):** A text input field with a masked key '*****'.
- Idioma dos documentos:** A dropdown menu with 'Português' selected.
- Concorrência:** A text input field with '4'.
- Tipos (separados por vírgula, vazio = todos):** A text input field with 'procedure,transaction,webpanel'.
- Limite (0 = todos):** A text input field with '0'.
- Caracteres do documento enviados ao modelo (reduza para 16.000 em um modelo local de contexto pequeno):** A text input field with '60000'.

Aba Narrativa

Conexões

Uma **conexão** é um provedor, um modelo e suas credenciais, guardados com um nome. Você pode ter várias (por exemplo, uma conta da empresa e um modelo local) e escolher qual usar em cada execução.

1. Em **Conexão**, clique em **+ Nova** e digite um nome.

- 2. Escolha o **Provedor**. O **Modelo** e a **URL base** são preenchidos com valores padrão que você pode alterar.
- 3. Cole a **API key desta conexão**, ou marque **Roda local** se você usa um modelo no seu computador ou na sua rede.
- 4. As alterações são salvas ao executar. **Excluir** remove a conexão selecionada.

Provedor	Modelo padrão	Observações
Anthropic (Claude)	claude-sonnet-5	Ótima qualidade de prosa e de respeito aos nomes
OpenAI (GPT)	gpt-4o-mini	
Google (Gemini)	gemini-3.8-flash	Contexto amplo e rápido; boa opção para KBs grandes
DeepSeek, Zhipu (GLM), Moonshot (Kimi), Alibaba (Qwen)	conforme o provedor	Aceitam Roda local se o modelo estiver na sua própria infraestrutura
Google (Gemma)	gemma3:latest	Pensado para rodar localmente
OpenRouter	um modelo do seu catálogo	Uma única chave para muitos modelos
mock (sem modelo)	—	Gera texto de teste, sem custo; serve para verificar o fluxo

Modelos locais

Com **Roda local**, a API key fica oculta e a **URL base** aponta para o seu servidor compatível com OpenAI: `http://localhost:11434/v1` para Ollama, `http://localhost:1234/v1` para LM Studio, ou o endereço de outra máquina da sua rede. Instale e baixe o modelo nesse servidor antes de executar (por exemplo `ollama pull qwen3.6`).

Com um modelo local **nenhum documento sai da sua rede**. A contrapartida é a velocidade e o tamanho do contexto: se o modelo tiver uma janela de contexto pequena, reduza **Caracteres do documento enviados ao modelo** para 16.000 ou menos.

Onde as chaves ficam guardadas

As API keys ficam guardadas no arquivo de configurações do UltraGX (`%APPDATA%\UltraGX\settings.json` no Windows, `~/.config/UltraGX/settings.json` no Linux e no macOS), dentro do seu perfil de usuário. Elas não são gravadas na pasta de saída nem enviadas ao UltraGX. Trate esse arquivo como um segredo.

Opções da execução

Opção	O que faz
Projeto	o projeto a narrar
Idioma	idioma da prosa (convém usar o mesmo do projeto)
Concorrência	objetos em paralelo. Com provedores na nuvem, 4 a 8 acelera bastante; com um modelo local, 1

Opção	O que faz
Tipos	tipos de objeto a narrar, separados por vírgula (por exemplo <code>procedure,transaction,webpanel</code>). Vazio = todos os tipos com lógica
Limite	quantidade máxima de objetos nesta execução (0 = todos). Útil para testar com 10 objetos antes de lançar a KB inteira
Caracteres do documento enviados ao modelo	tamanho máximo de cada documento enviado (60.000 por padrão). Documentos mais longos são cortados

Com **Tipos** vazio, o UltraGX **não narra as declarações**: atributos, domínios, SDTs, grupos de subtipos, objetos externos, data views e configurações de padrões. O documento determinístico delas já diz tudo o que há a dizer, e narrá-las só gasta tokens. Se mesmo assim quiser narrar alguma, escreva-a em **Tipos**.

Executar, acompanhar e retomar

Clique em **Executar / retomar**. O log mostra cada objeto com seu resultado. Cada resposta é incorporada ao seu documento **na hora**: você pode abrir o objeto no **Visualizador** (com **Recarregar**) enquanto a execução continua, e o visualizador completo é regenerado periodicamente e ao final.

Se você parar a execução ou ela for interrompida, clique de novo em **Executar / retomar**: os objetos já narrados não são enviados novamente.

Resultados por objeto

Status	O que significa	É publicado
ok	a prosa passou na validação	sim, na seção Significado de negócio e nas seções correspondentes
degraded	depois das novas tentativas, o modelo continuou citando nomes que não existem no objeto	não; o documento mostra um aviso com os nomes inventados
error	o provedor falhou (chave inválida, cota esgotada, resposta inválida, tempo esgotado)	não

Repetir degradados / erros processa novamente só esses objetos. Os erros transitórios do provedor (limites de taxa, erros 5xx, quedas de rede) já são repetidos automaticamente com espera progressiva durante a execução.

O status de todos os objetos está na página **Narrativa** do visualizador e na coluna **Narrativa** da lista de objetos.

Custos e privacidade

- Cada objeto é uma chamada ao modelo com o seu documento completo (até o limite de caracteres). Para estimar o custo, execute com **Limite 20**, veja os tokens que o log informa no final e extrapole para a quantidade de objetos com lógica da sua KB.
- Com provedores na nuvem, o documento de cada objeto (incluindo o seu código GeneXus) é enviado a esse provedor com a sua conta e sob as condições dele. Se a KB tiver restrições de confidencialidade, use um modelo local ou um provedor aprovado pela sua organização.

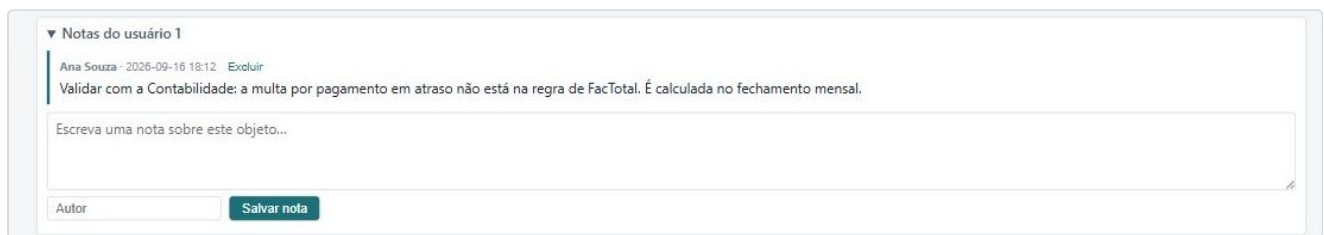
- Na edição demo, só os objetos da amostra são narrados.

Notas do usuário

As notas guardam o que a análise não tem como saber: que um procedimento não é mais usado, que uma regra atende a uma norma, quem é o responsável funcional de um módulo ou o que a equipe decidiu para a migração. Elas ficam dentro da documentação, então um agente de IA também as lê.

As notas são adicionadas **pelo aplicativo** (aba **Visualizador**). Se você abrir o visualizador em um navegador separado, elas aparecem, mas não podem ser criadas nem excluídas.

Nota sobre um objeto

A interface mostra uma seção intitulada "Notas do usuário 1". Abaixo do título, há uma linha de texto com o nome "Ana Souza", a data "2026-09-16 18:12" e um link "Excluir". Segue-se o texto da nota: "Validar com a Contabilidade: a multa por pagamento em atraso não está na regra de FacTotal. É calculada no fechamento mensal." Abaixo disso, há uma caixa de texto com o placeholder "Escreva uma nota sobre este objeto...". Na base da interface, há um campo "Autor" e um botão "Salvar nota".

Notas na página de um objeto

1. Abra o objeto no **Visualizador**.
2. Em **Notas do usuário**, escreva a nota e, se quiser, o seu nome em **Autor**.
3. Clique em **Salvar nota**.

A nota aparece imediatamente na página e fica registrada com data e autor. **Excluir** a apaga.

Nota sobre vários objetos de uma vez

1. Em **Objetos**, filtre a lista (por exemplo, por pasta ou por cluster).
2. Marque as caixas da última coluna, ou a caixa do cabeçalho para **Selecionar visíveis**.
3. Na barra que aparece acima, escreva a nota e clique em **Adicionar nota aos objetos selecionados**.

Cada objeto recebe sua cópia da nota. A coluna **Notas** da lista indica quais objetos têm notas.

Onde ficam

- Em `annotations/` dentro do projeto, um arquivo JSON por objeto. É a fonte da verdade: copie essa pasta se quiser levar as notas para outra máquina.
- No documento Markdown do objeto, seção **Notas do usuário**.
- No JSON do objeto, campo `user_notes`.

Quando você executa novamente o pipeline sobre o mesmo projeto (por exemplo, com um export mais recente da KB), as notas são reincorporadas aos documentos regenerados. Se um objeto sumiu da KB, o arquivo de notas dele continua em `annotations/`, mas não é mais exibido.

Migrar com um agente de IA

A saída do UltraGX foi escrita para que um agente de programação (Claude Code, Codex, Cursor, Antigravity ou outro) possa reconstruir o aplicativo em outra tecnologia sem abrir o GeneXus. O agente não precisa interpretar código GeneXus do zero: ele lê regras já explicadas, pseudocódigo por evento, o SQL implícito de cada `For each`, o modelo de dados com suas chaves e uma ordem de reconstrução.

Este capítulo descreve um método de trabalho que funciona bem. Não é a única forma, mas evita os erros mais comuns: pedir tudo de uma vez ao agente, ou deixar que ele invente estrutura que a KB já define.

1. Preparar o espaço de trabalho

1. Gere a documentação com a **edição completa** e, se possível, com a narrativa em `ok` para os objetos de negócio. Use o idioma em que a sua equipe trabalha.
2. Crie o repositório do novo aplicativo **separado** da saída do UltraGX.
3. Deixe a saída acessível ao agente como material somente leitura: por exemplo, uma pasta irmã (`../kb-docs/`) ou um submódulo. Não é preciso copiar `viewer/`, `_pipeline/` nem `_source_text/`.
4. No novo repositório, escreva o arquivo de instruções do agente (`CLAUDE.md`, `AGENTS.md` ou as regras do Cursor) com:
 - a tecnologia de destino e suas convenções (linguagem, framework, banco de dados, estrutura de pastas);
 - a localização da documentação da KB e a indicação de ler primeiro o seu `README.md`;
 - a regra de que os nomes de tabelas, atributos e regras vêm da documentação, não são inventados;
 - que as **notas do usuário** prevalecem sobre o que foi inferido.

Exemplo de bloco para esse arquivo (se os seus documentos estiverem em português, as seções se chamam **Notas do usuário** e **Limites desta análise**):

```
## 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. Ordem de leitura

O `README.md` da saída indica ao agente como percorrê-la. Em resumo:

1. `README.md`, `kb_summary.md` e `kb_clusters_summary.md`: o que existe e como está agrupado.
2. `datamodel/index.md`: tabelas, chaves e qual transação define cada tabela.
3. `global_rebuild_roadmap.md`: em que ordem reconstruir.
4. Para cada cluster: `clusters/cluster_NNN.knowledge.md`, `.flow.md` e `.rebuild.md`.
5. Para cada objeto: o seu documento em `objects/` (caminho em `kb_index.json` → `doc_path`).
6. Para seguir relações: `kb_graph.json` ou a seção **Referências cruzadas** de cada documento.

3. Trabalhar por fases e por cluster

1. **Modelo de dados primeiro.** Peça ao agente o esquema do banco de dados e as entidades a partir de `datamodel/`. Revise-o contra o banco real antes de seguir: todo o resto se apoia nisso.
2. **Siga o roteiro.** Pegue a primeira fase de `global_rebuild_roadmap.md` e, dentro dela, um cluster por vez.
3. **Um cluster, uma tarefa.** Para cada cluster, o agente lê o seu plano de reconstrução e os documentos dos seus objetos, propõe o design, implementa e escreve testes a partir das regras de negócio.
4. **Feche o cluster** antes de passar ao próximo: testes passando, revisão das regras uma a uma e notas do usuário com as decisões tomadas (elas ficam para as próximas sessões do agente).

4. Pedidos de exemplo

Orientação

Leia `../kb-docs/README.md`, `kb_summary.md` e `kb_clusters_summary.md`. Resuma o que o aplicativo faz, suas áreas funcionais e as 10 tabelas mais importantes. Não escreva código ainda.

Modelo de dados

A partir de `../kb-docs/datamodel/`, gere as migrations do banco de dados para PostgreSQL e as entidades do ORM. Respeite nomes, tipos e chaves primárias e estrangeiras. Liste qualquer tabela cuja definição esteja incompleta.

Um cluster

Implemente o cluster `cluster_004` conforme `clusters/cluster_004.rebuild.md`. Leia o documento de cada objeto membro e reproduza suas regras de negócio e o SQL implícito dos seus `For each`. Para as telas, use a seção de interface do usuário. Escreva um teste por regra de negócio e marque o que você não conseguir reproduzir com certeza.

Um objeto específico

Procure em `kb_index.json` o procedure `CalcularDescuento`, leia o seu documento e reimplimente-o como serviço. Explique

cada ramo do pseudocódigo em um comentário do teste correspondente.

Verificação

Compare o serviço InvoiceService com objects/Facturacion/Transaction.Factura.md: enumere cada regra de negócio do documento e indique onde ela está implementada ou se está faltando.

5. Boas práticas

- **Contexto delimitado.** Não peça ao agente que leia a pasta inteira: com `kb_index.json` e `doc_path` ele encontra o que precisa para cada tarefa.
- **Confiança.** Priorize a revisão humana dos objetos com `confidence_level` baixo e do que aparece em **Limites desta análise**.
- **Objetos de biblioteca e de padrões.** Os objetos marcados como biblioteca (módulos importados do GeneXus ou de terceiros) e a estrutura gerada por padrões (Work With, K2BTools) não são migrados tal como estão: são substituídos pelos equivalentes do framework de destino.
- **A KB continua viva.** Se o aplicativo GeneXus receber mudanças durante a migração, exporte e execute de novo o pipeline sobre o mesmo projeto; as notas são preservadas e as diferenças ficam visíveis no controle de versão se a saída estiver versionada.

Linha de comando

Tudo o que o aplicativo faz é executado pelo motor, `ultragx-engine`, que também pode ser usado diretamente em um terminal: para automatizar a documentação de várias KBs, agendá-la em um servidor ou integrá-la a um processo de integração contínua. Ele usa a mesma licença do aplicativo.

Onde está o motor

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

Nos exemplos é usado `ultragx-engine`; substitua pelo caminho correspondente ou adicione a pasta ao `PATH`.

Comandos

version e machine-id

```
ultragx-engine version      # versão do motor
ultragx-engine machine-id   # id da máquina, para pedir uma licença vinculada a ela
```

license

```
ultragx-engine license      # status da licença, em JSON
ultragx-engine license activate --code ULGX-XXXX-XXXX  # ativa uma licença com um código
```

A demo também pode ser pedida sem o aplicativo, em duas etapas:

```
ultragx-engine license register --first-name Ana --last-name Souza --company "Minha Empresa Ltda." \
  --email ana.souza@minhaempresa.com.br --gx-version 18 --kb-size 3000 --lang pt-BR
ultragx-engine license confirm --email ana.souza@minhaempresa.com.br --code 123456
```

pipeline

Documenta uma KB a partir de um export.

```
ultragx-engine pipeline --source xpz --input C:/exports/MinhaKB.xpz --out D:/ultragx/MinhaKB
--lang pt-BR
ultragx-engine pipeline --source gxttext --input D:/exports/MinhaKB_texto --out
D:/ultragx/MinhaKB --lang pt-BR --no-ref
```

Opção	Descrição
<code>--source</code>	<code>xpz</code> (arquivo <code>.xpz</code> ou <code>.xml</code>) ou <code>gxttext</code> (pasta com <code>src/</code> e <code>ref/</code>)
<code>--input</code>	arquivo ou pasta de entrada
<code>--out</code>	pasta do projeto (é criada se não existir)
<code>--lang</code>	idioma dos documentos: <code>en</code> , <code>es</code> ou <code>pt-BR</code>
<code>--no-ref</code>	com <code>gxttext</code> , ignora os objetos de módulos referenciados
<code>--no-viewer</code>	não gera o visualizador (mais rápido; pode ser gerado depois)

O progresso é escrito na saída de erro e, ao terminar, o resumo da execução sai em JSON na saída padrão.

export

Abre uma KB com o GeneXus instalado (somente Windows, com o conector), a exporta para texto e executa o pipeline.

```
ultragx-engine export --kb-path "C:/Models/MinhaKB" --kb MinhaKB --out D:/ultragx/MinhaKB --
lang pt-BR
```

Opção	Descrição
<code>--kb-path</code>	pasta da KB (a que contém o <code>.gxw</code>)
<code>--kb</code>	nome curto da KB
<code>--out</code>	pasta do projeto
<code>--gx-version</code>	força uma instalação do GeneXus (<code>18</code> , <code>17</code> , <code>XEv3</code> ...); por padrão é deduzida da KB
<code>--names</code>	exporta só alguns objetos: nomes ou seletores como <code>type:Transaction</code>
<code>--no-reference</code>	ignora os módulos referenciados
<code>--skip-pipeline</code>	só exporta, sem documentar
<code>--timeout</code>	tempo máximo da exportação em segundos (3600 por padrão)
<code>--lang</code> , <code>--no-viewer</code>	igual a <code>pipeline</code>

viewer

Regenera o visualizador de um projeto sem analisar a KB de novo.

```
ultragx-engine viewer --out D:/ultragx/MinhaKB --lang pt-BR
```

narrative

Executa a narrativa com IA sobre um projeto já documentado. Na linha de comando, as credenciais são obtidas de variáveis de ambiente: `ANTHROPIC_API_KEY` para a Anthropic, `OPENAI_API_KEY` (e `OPENAI_BASE_URL` para outro servidor) para os provedores compatíveis com OpenAI, incluindo Gemini, OpenRouter e os modelos locais.

```
export OPENAI_API_KEY=...          # no PowerShell: $env:OPENAI_API_KEY = "..."
ultragx-engine narrative run --out D:/ultragx/MinhaKB --provider openai \
  --base-url https://generativelanguage.googleapis.com/v1beta/openai/ --model gemini-3.8-
  flash \
  --lang pt-BR --concurrency 8
```

Opção	Descrição
<code>--provider</code>	<code>anthropic</code> , <code>openai</code> (qualquer servidor compatível) ou <code>mock</code>
<code>--model</code> , <code>--base-url</code>	modelo e URL do servidor
<code>--lang</code>	idioma da prosa
<code>--concurrency</code>	objetos em paralelo (4 por padrão)
<code>--types</code>	só estes tipos, separados por vírgula
<code>--all-types</code>	inclui também as declarações (atributos, domínios, SDTs...)
<code>--only</code>	só estes objetos, por <code><Tipo>.<Nome></code> separados por vírgula
<code>--limit</code>	quantidade máxima de objetos
<code>--no-resume</code>	processa novamente objetos já narrados
<code>--max-doc-chars</code>	caracteres do documento enviados (60.000 por padrão; 16.000 com um servidor local se não for indicado)
<code>--no-source</code>	não envia o código GeneXus original ao modelo
<code>--timeout</code>	segundos de espera por resposta (300 por padrão)
<code>--viewer-every</code>	segundos entre regenerações do visualizador durante a execução (600 por padrão; 0 = só no final)

notes

Adiciona ou exclui notas do usuário sem o aplicativo.

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

`nota.json` tem o formato `{"stems": ["Procedure.CalcularDescuento"], "text": "...", "author": "Ana"}`. `apply` reincorpora todas as notas aos documentos.

Códigos de saída

Código	Significado
0	terminou bem
1	a narrativa terminou com objetos em <code>error</code>
2	uso incorreto (comando ou opções inválidos) ou, em <code>export</code> , não foi possível abrir a KB
3	licença ausente, inválida, vencida ou revogada

Exemplo: documentar todas as KBs de uma pasta

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

Solução de problemas e perguntas frequentes

Instalação e inicialização

O Windows mostra "O Windows protegeu o computador". É o SmartScreen com um instalador que ainda tem pouca reputação. Clique em **Mais informações → Executar assim mesmo**. Se a sua organização bloquear essa opção, peça ao suporte técnico que libere o instalador.

A janela do UltraGX abre em branco ou não abre (Windows). O runtime Microsoft Edge WebView2 está faltando ou danificado. Instale-o a partir da página da Microsoft ("WebView2 Runtime", instalador *Evergreen*) e abra o UltraGX de novo. O instalador do UltraGX o adiciona automaticamente.

O aplicativo não abre no Linux. Execute-o em um terminal (`ultragx` ou a ApplImage) para ver o erro. Se ele mencionar `libwebkit2gtk`, instale `libwebkit2gtk-4.1-0` (Debian/Ubuntu) ou `webkit2gtk4.1` (Fedora).

O macOS diz que o UltraGX "não pode ser aberto". A versão para macOS não é assinada com um Developer ID da Apple. Dê dois cliques em **Allow UltraGX.command** na imagem de disco depois de arrastar o UltraGX para Aplicativos (ou execute `xattr -cr /Applications/UltraGX.app` no Terminal), ou autorize-o em **Ajustes do Sistema → Privacidade e Segurança → Abrir Mesmo Assim**.

Licença

Aparece "Não foi possível contatar o serviço de licenças". O computador não alcança `https://licenses.ultra-gx.com`. Verifique a conexão com a internet; em redes corporativas, peça que liberem HTTPS para esse domínio (também pelo proxy). As demais mensagens do registro estão explicadas em [Primeiro início e licença](#).

A execução termina com "A valid license file (ultragx.lic) is required..." ou código 3. A licença está ausente, venceu ou foi revogada. Confira **Configurações → Licença**. Se a demo terminou, escreva para nós para passar à edição completa.

Na demo, nenhum objeto saiu detalhado. A execução não conseguiu consultar o serviço de licenças e continuou sem amostra. O motivo está no aviso do `README.md` e do visualizador. Resolva a conexão e execute o pipeline de novo.

Troquei de computador. A licença fica vinculada à máquina onde foi ativada. Escreva para nós com o novo **Id desta máquina** para transferi-la.

Documentar uma KB

A fonte "live" aparece desabilitada. **Configurações → Ambiente** mostra o motivo: *Conector GeneXus: não encontrado* (Linux, macOS ou uma cópia sem conector) ou *GeneXus instalado: nenhum detectado*. Nesses casos, exporte a KB para XPZ na máquina onde está o GeneXus.

A exportação ao vivo não avança ou falha ao abrir a KB. A KB está aberta na IDE do GeneXus ou em outra ferramenta: feche-a e execute de novo. Se o log repetir o mesmo objeto por vários minutos, o conector continua extraindo um objeto grande; em KBs de milhares de objetos a exportação pode levar mais de uma hora.

Escolhi a versão errada do GeneXus. Deixe **auto (a partir da KB)**. Force uma instalação só se a KB abrir com uma versão diferente da que é detectada.

Alguns SDTs ou transações aparecem sem estrutura. Se a KB foi documentada com uma versão anterior à 0.2.5, execute-a de novo com a versão atual (com a fonte live, também a exportação).

A aceitação terminou em warning . A documentação foi gerada completa; o controle de qualidade encontrou algo a revisar. O detalhe está em **Aceitação**, no visualizador. **failed** indica um problema que convém relatar.

O resumo avisa library_flag_suspect . Quase todos os objetos foram marcados como biblioteca, o que indica um export que não separou bem os objetos próprios dos referenciados. Exporte a KB de novo (com o conector atual, se for ao vivo).

Os caminhos ou os nomes de pasta são muito longos (Windows). A saída replica os módulos e pastas da KB. Use uma pasta de saída curta, como **D:\ultragx** .

A pasta de saída está no OneDrive / Dropbox e tudo fica lento. A sincronização processa milhares de arquivos ao mesmo tempo. Use uma pasta fora das pastas sincronizadas.

Narrativa

Mensagem no log	Causa	Solução
401 , 403 , invalid API key	a chave da conexão não é válida ou não tem permissão para esse modelo	confira a API key da conexão
404 , model not found	o nome do modelo não existe nesse provedor	corrija Modelo com um nome do catálogo do provedor
429 , quota, rate limit	limite de uso do provedor	o motor tenta de novo sozinho; se persistir, reduza Concorrência ou confira a cota da sua conta
connection refused com um modelo local	o servidor local não está rodando ou a URL não é a correta	inicie o Ollama / LM Studio e verifique a URL base
muitos degraded	o modelo cita nomes que não existem	teste um modelo mais capaz; com modelos locais, aumente o tamanho do contexto

A prosa não aparece no visualizador. Clique em **Recarregar**. Só os objetos em **ok** são publicados.

Onde olhar

O quê	Onde
Log de uma execução	Tarefas → Log enquanto o aplicativo está aberto
Resultado da execução	<code>run_summary.json</code> e <code>acceptance_report.md</code> na pasta do projeto

O quê	Onde
Status da narrativa	<code>narrative/checkpoint.json</code> e um JSON por objeto em <code>narrative/</code>
Configurações e licença	<code>%APPDATA%\UltraGX</code> (Windows) ou <code>~/.config/UltraGX</code> (Linux e macOS)

Suporte

Escreva para contact@ultra-gx.com com:

- a versão do UltraGX (**Configurações → Ambiente → Aplicativo**) e o sistema operacional;
- a mensagem de erro ou o final do log da tarefa;
- `run_summary.json` e `acceptance_report.md` do projeto, se o problema for de uma execução.

Não envie o XPZ, o código da KB nem a pasta de saída completa, a menos que pedirmos e a sua organização autorize.

Perguntas frequentes

O UltraGX modifica a minha KB? Não. Ele só lê: o conector não tem nenhum caminho de escrita e, com um XPZ, nem sequer toca na KB.

Preciso ter o GeneXus instalado? Só para a fonte live. Com um XPZ ou um export de texto, não.

Quais versões do GeneXus são suportadas? Da 7.5 à 18 por XPZ/XML; do X Evolution 1 ao 18 ao vivo.

Posso documentar a mesma KB muitas vezes? Sim. Com o mesmo nome de projeto, ela é regenerada na mesma pasta e as notas e a narrativa são preservadas. Na demo, a amostra de 25 objetos é sempre a mesma.

Posso compartilhar a documentação? Sim: a pasta do projeto é autocontida e o visualizador funciona sem conexão. Lembre-se de que ela contém o código da KB.

O código-fonte é enviado para algum lugar? Só se você usar a narrativa com um provedor na nuvem, e para esse provedor. Veja [Narrativa com IA](#).

Glossário

Termo	Significado
Aceitação	Controle de qualidade que o motor executa ao final de cada pipeline: <code>passed</code> , <code>warning</code> ou <code>failed</code> .
Anexo determinístico	Última parte de cada documento de objeto, com a evidência técnica completa da análise.
Árvore de chamadas	Diagrama de quem chama quem a partir de um objeto ou de um ponto de entrada.
Cluster	Grupo de objetos que trabalham juntos em uma mesma área funcional, inferido do grafo de dependências e da localização na KB.
Conector	Componente do UltraGX para Windows que abre uma KB com o GeneXus instalado, em modo somente leitura, e a exporta para texto.
Confiança	Pontuação de 0 a 100 (<code>high</code> , <code>medium</code> , <code>low</code>) que indica quanta evidência sustentou a análise de um objeto.
Conexão	Provedor, modelo e credenciais de IA guardados com um nome, para a narrativa.
Declaração	Objeto que apenas define um tipo, uma estrutura ou uma configuração (atributo, domínio, SDT, objeto externo, configuração de padrão). Não é narrado por padrão.
<code>degraded</code>	Resultado da narrativa de um objeto quando o modelo citou nomes que não existem nele. Não é publicado.
Diagrama de fluxo de dados (DFD)	Diagrama de quais objetos leem e gravam quais tabelas.
Edição demo / completa	Tipos de licença. A demo dura 10 dias e detalha 25 objetos; a completa documenta tudo sem conexão.
Export de texto	Formato do GeneXus 17 e 18 que guarda cada objeto como arquivo de texto, nas pastas <code>src/</code> e <code>ref/</code> .
Roteiro de reconstrução	Ordem sugerida para reconstruir o aplicativo, por fases, conforme as dependências entre clusters.
KB (Knowledge Base)	A base de conhecimento do GeneXus: todos os objetos de um aplicativo.
Knowledge pack	Resumo estruturado de um objeto: compreensão funcional, modelo de interação, uso de dados, dependências, riscos e lacunas.
Live (fonte)	Leitura da KB diretamente com o GeneXus instalado, por meio do conector.
Motor	<code>ultragx-engine</code> , o programa que analisa a KB e escreve a documentação. O aplicativo o executa por você.
Narrativa	Prosa escrita por um modelo de linguagem e validada contra os fatos do objeto.
Notas do usuário	Comentários que a equipe adiciona aos objetos; são preservados entre execuções e prevalecem sobre o que foi inferido.
Objeto de biblioteca	Objeto de um módulo importado (do GeneXus ou de terceiros), não desenvolvido na KB.
Objeto de negócio	Transações, Web Panels, Work Panels, Procedimentos, Relatórios, Query Objects e SDTs: onde vive a lógica do aplicativo.

Termo	Significado
Perfil funcional	Classificação do que um objeto faz (por exemplo, manutenção de dados, consulta, processamento em lote).
Pipeline	A execução completa do motor sobre uma KB: leitura, análise, relações e escrita.
Projeto	A pasta de saída de uma KB documentada.
Pseudocódigo	Tradução legível de cada evento e sub-rotina do objeto.
SQL implícito	A consulta que o GeneXus gera para um <code>For each</code> , reconstruída pelo motor: tabela base, joins, condições e ordem.
Stub	Na demo, documento resumido de um objeto fora da amostra.
Visualizador	Site estático gerado para cada projeto, navegável sem conexão.
XPZ	Arquivo de exportação de objetos do GeneXus (XML compactado).