O futuro do desenvolvimento de software

Do código ao conhecimento: a nova arquitetura sobre a qual será construído o software empresarial



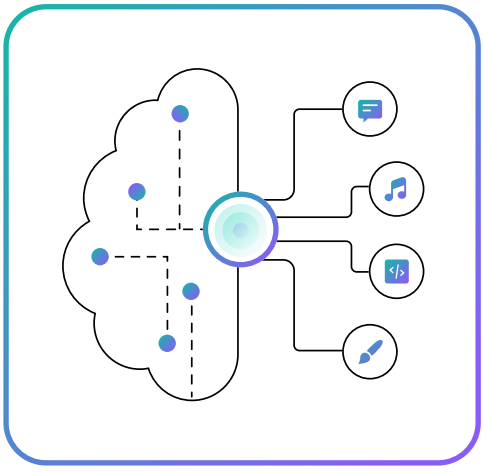
Estamos vivendo uma dupla revolução tecnológica.

Duas forças simultâneas estão se potencializando mutuamente e redefinindo a maneira como o software empresarial é construído, operado e concebido.

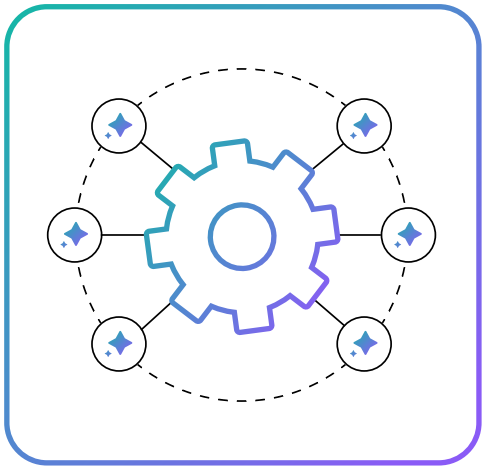
A primeira revolução é a da inteligência artificial generativa. Desde o lançamento do ChatGPT em novembro de 2022, a capacidade de gerar código, processar linguagem natural e se comunicar com sistemas por meio da linguagem humana deixou de ser experimental para se tornar o novo normal. Pela primeira vez na história do software, é possível programar sistemas em massa declarando intenção em vez de escrever instruções.

A segunda revolução é a agêntica, e é a que recebe menos atenção da imprensa, apesar de ser igual ou mais significativa que a primeira. Os sistemas construídos hoje contam com agentes que acessam o conhecimento, raciocinam sobre ele e agem de forma autônoma. Essa revolução emerge sobre os alicerces da primeira e define uma nova forma de operar as organizações.

Nicolás Jodal, CEO da GeneXus, ressalta que é a primeira vez na história que duas revoluções tecnológicas ocorrem simultaneamente.



Revolução GenAI



Revolução Agêntica

As últimas grandes transformações do setor chegaram em momentos distintos e foram sentidas em camadas: o smartphone e a conectividade móvel redefiniram a relação entre as pessoas e a tecnologia a partir de 2007, criando indústrias inteiras que não existiam e destruindo outras que pareciam inabaláveis. Seu impacto foi imediato e visível para qualquer pessoa. Depois, à medida que as organizações assimilavam essa mudança, chegou a nuvem: **Amazon Web Services, Google Cloud e Microsoft Azure** transformaram radicalmente a forma como o software empresarial é construído e operado, passando de uma infraestrutura própria, cara e difícil de escalar, para um serviço sob demanda, acessível e elástico. Seu impacto foi mais profundo, porém mais lento, e dominou a agenda do mundo empresarial por mais de uma década.

Em cada um desses casos, as organizações tiveram tempo de entender a mudança, se reorganizar e se adaptar antes que a próxima chegasse. Hoje essa margem não existe. A revolução generativa e a revolução agêntica não se sucedem, elas se sobrepõem. E o que torna essa sobreposição qualitativamente diferente de qualquer mudança anterior é que cada uma amplifica a outra: os agentes são mais poderosos porque a IA generativa existe, e a IA generativa é mais útil porque existem os agentes que a orquestram. O resultado é a multiplicação de ambas as mudanças. Essa é a magnitude do que está acontecendo.



*Neste whitepaper exploramos essa transformação por meio da visão de Gastón Milano, **CTO da GeneXus** e de **Glob.AI na Globant**, que analisa as mudanças que estão ocorrendo no desenvolvimento de software empresarial, os papéis que estão emergindo, os ativos que se tornam estratégicos e as decisões que as organizações devem tomar hoje para se posicionar corretamente diante dessa revolução.*

Que software é preciso construir?

Antes de falar sobre velocidade, agentes, geração de código e como construir software nesta nova era, é importante entender que tipo de software deve ser construído. Essa distinção é, segundo Gastón Milano, um dos erros mais comuns do mercado, já que, enquanto o "o quê" não for respondido, qualquer conversa sobre o "como" carece de fundamento.

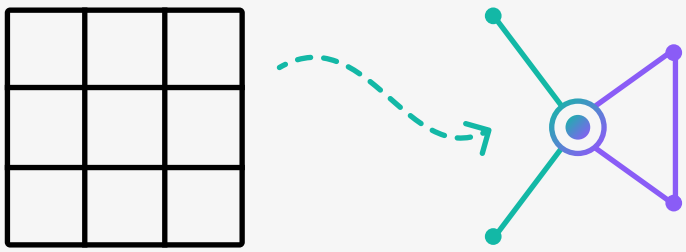
O que vem pela frente é muito mais desafiador, como afirmam os líderes globais:



Satya Nadella (Chairman e CEO da Microsoft)

Ele descreveu um futuro em que toda a lógica de negócio hoje existente como código estático passará a ser operada por agentes de inteligência artificial, restando apenas alguns sistemas de registro, enquanto tudo o que hoje conhecemos como SaaS e software de serviço está passando por uma profunda reinvenção.

"Quando os líderes da transformação tecnológica global fazem afirmações disruptivas, a reação inicial costuma ser o ceticismo. Mas a história mostra que, à medida que as mudanças vão se concretizando, fica claro que eles não estavam errados", acrescenta Milano.



O paradigma agêntico: uma nova economia

Existe toda uma infraestrutura de protocolos e padrões, nenhum dos quais é inteligência artificial generativa, mas sim engenharia pura, que está amadurecendo e definindo o que significa construir software agêntico.

"Hoje é possível verificar se o que você construiu é verdadeiramente agêntico ou não. Isso reflete que o conceito de agêntico já tem uma definição técnica concreta, verificável e auditável. Esse amadurecimento de protocolos nos lembra o período de meados dos anos 90, quando a internet começou a consolidar sua pilha de protocolos. Da mesma forma, hoje está amadurecendo a infraestrutura sobre a qual o mundo agêntico vai operar, e essa consolidação é o que faz com que o conceito de agêntico deixe de ser uma aspiração para se tornar algo tecnicamente verificável e auditável."

Esses protocolos incluem:



Protocolos de descoberta de agentes
Mecanismos para que um agente encontre outro disponível no ecossistema.



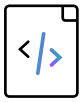
Protocolos de compra e venda
Novas formas de transação econômica entre agentes e sistemas.



Protocolos de uso de ferramentas
Padrões para que os agentes acessem e usem capacidades externas de forma estruturada.



Protocolos de navegação web
Mecanismos para que os agentes interajam com interfaces web de maneira autônoma.



Padrões de formato de saída
Definições que permitem maior interoperabilidade e dinamismo entre sistemas.

A queda do paradigma tradicional de software

Concordemos ou não, vários dos pilares do desenvolvimento de software como o conhecemos estão se tornando obsoletos.

Um aplicativo é, em essência, uma previsão. Alguém decidiu de antemão o que a maioria dos usuários precisaria e fixou isso em uma tela. Um dashboard é uma aposta sobre quais visualizações serão úteis. Uma página web é um design pensado para o usuário médio. Nenhum deles se adapta a quem realmente o usa.

"A engenharia de software resolveu isso da única maneira que podia: construindo para a média. Funcionou, mas sempre foi uma solução de compromisso. A personalização real, aquela que a indústria buscou por décadas, nunca foi alcançável com esse modelo, porque preparar algo diferente para cada pessoa era inviável. Os sistemas conversacionais generativos rompem essa lógica. Eles não têm uma resposta fixa para todos: geram uma resposta diferente para cada pessoa, com base no contexto específico que têm sobre ela. Duas pessoas que fazem a mesma pergunta podem receber respostas completamente diferentes, e ambas podem estar corretas para cada uma delas", explica Milano.

Quando a velocidade de geração se torna indistinguível da de uma conversa humana, a previsão pré-montada deixa de ter vantagem. Nesse momento, personalizar não custa mais do que padronizar, e a média deixa de ser o único caminho possível.

Para onde vamos?

A experiência do usuário está migrando para interfaces em que o usuário simplesmente pede a um agente o que precisa, e esse agente renderiza isso no formato adequado, áudio, vídeo, texto, livro, em tempo real.

A tabela a seguir resume as mudanças mais significativas no que está perdendo e ganhando relevância neste novo paradigma:

O QUE PERDE RELEVÂNCIA	O QUE GANHA RELEVÂNCIA
Múltiplos aplicativos	Agentes conversacionais como ponto único de acesso
Interfaces de formulários e telas estáticas	Interfaces 100% dinâmicas renderizadas em tempo de execução
Dashboards e visualizações predefinidas	Visualizações geradas sob medida conforme o contexto do usuário
Lógica de negócio codificada estaticamente	Lógica de negócio operada por agentes sobre bases de conhecimento
CRUD e telas fixas	Agentes acessando conhecimento em tempo real
Pull Requests e revisão linha a linha	Supervisão arquitetônica e validação a posteriori
IDEs tradicionais	Orquestradores de frotas de agentes

Gastón Milano identifica sinais concretos de para onde essa transformação está indo:

1. "O modelo que vem por aí não tem formulários, não tem telas fixas, não tem CRUD. Há agentes que acessam o conhecimento e geram em tempo real exatamente o que cada pessoa precisa, como se a Netflix renderizasse o conteúdo no momento em que você o pede, em vez de tê-lo pré-gravado."

2. "Para que isso funcione bem, é preciso dar a esses agentes mais contexto, mais conhecimento declarado sobre como devem se comportar, quais regras seguir e como servir melhor a cada cliente. Isso inclui políticas, processos, papéis, regras de decisão. Tudo aquilo que hoje vive na cabeça das pessoas, em documentos dispersos ou em convenções não escritas, precisa estar codificado para que os agentes possam agir com base nisso."

3. "Quando os agentes conseguem agir com esse conhecimento, a empresa adquire propriedades novas: tudo o que acontece se torna versionável, auditável, derivável em direção a caminhos diferentes, operável de forma autônoma e, ao mesmo tempo, compreensível para os humanos que precisam supervisioná-lo."

4. "Isso não é uma ideia isolada. Pesquisadores como **Andrej Karpathy** (ex-Diretor de IA na Tesla e membro fundador da OpenAI), entre outros, estão convergindo na mesma direção: as empresas do futuro não são programadas, são documentadas. O conhecimento organizacional deixa de ser implícito e passa a ser um ativo explícito, vivo e executável."

Companies as Code:
o novo paradigma organizacional

Para Gastón Milano, é fundamental tratar as empresas como se trata o software: da mesma forma que o código de um sistema pode ser versionado, implantado e executado, o conhecimento de uma organização pode ser codificado e se tornar o motor sobre o qual operam os agentes de inteligência artificial.

Milano chama essa visão de Companies as Code: uma forma de entender a empresa em que todo o seu conhecimento organizacional, suas políticas, processos, papéis e decisões, está documentado, versionado e disponível para que os agentes possam operá-lo.

Em conjunto, isso representa uma mudança de era: passar de empresas que operam com conhecimento implícito, disperso na cabeça de seus funcionários ou enterrado em documentos, para empresas cujo conhecimento é explícito, vivo e executável. Quando esse conhecimento está corretamente codificado, obtêm-se propriedades que antes era impossível ter todas juntas:

Versionável
Cada mudança em uma política ou processo fica registrada. É possível saber o que dizia antes, o que diz agora e por que mudou.

Auditável
Tudo o que acontece na empresa é rastreável. Não há decisões que ocorram fora do sistema.

Derivável
A partir desse conhecimento base, é possível explorar caminhos alternativos, simular cenários ou tomar decisões diferentes com fundamento.

Operável por agentes
Os agentes não apenas consultam esse conhecimento, eles o executam. Podem agir em nome da empresa seguindo suas próprias regras documentadas.

Compreensível para humanos
Nada disso serve se apenas as máquinas o entenderem. O sistema precisa poder se explicar e ser supervisionado pelas pessoas.

O que é preciso fazer?

A transição para uma arquitetura agêntica não acontece em um único movimento. Milano identifica três passos concretos que as organizações devem dar para se posicionar corretamente nesta nova era.

Passo 1: Codificar o conhecimento

O conhecimento mais valioso de uma empresa não vive em bancos de dados relacionais nem em código. Vive em uma wiki: um conjunto de ideias conectadas entre si por meio de uma ontologia, ou seja, uma estrutura que define como os conceitos se relacionam. Quando esse conhecimento está codificado nesse formato, ele se torna operável. Os agentes podem navegá-lo, interpretá-lo e agir sobre ele. As pessoas podem se combinar com esses agentes trabalhando sobre a mesma base. A empresa deixa de depender de que o conhecimento esteja na cabeça de alguém e passa a tê-lo como um ativo real.

"A **GeneXus** é um bom exemplo disso, porque opera assim. Sua wiki interna contém desde como fazer uma boa apresentação até como responder perguntas difíceis, como conduzir reuniões, como cuidar da saúde. Nesta nova era, esses ativos se tornam consumíveis, expansíveis e escaláveis por agentes, de formas que antes não eram possíveis."

Para quem trabalha no ecossistema GeneXus, isso não é novidade: a Knowledge Base (KB) sempre foi um ativo central. O que muda é que agora esse ativo pode se conectar diretamente com a camada de execução. O conhecimento da empresa não apenas é documentado, ele é executado.

"A grande maioria da indústria começa do zero. Não tem políticas codificadas, não tem processos versionados, não tem conhecimento organizacional em um formato que os agentes possam consumir. Quem já tem uma KB da GeneXus tem algo que o resto ainda não sabe como construir. Essa distância não é pequena: é a diferença entre já ter o ativo estratégico do próximo ciclo tecnológico e estar apenas começando a entender o que é."

Nesse contexto, as KBs apresentam características que as tornam especialmente valiosas no mundo agêntico:

São versionáveis
Assim como o código, suas mudanças históricas podem ser rastreadas.

São consumíveis por agentes
Com o GeneXus for Agents, as KBs podem ser acessadas e utilizadas diretamente por agentes de IA.

São guiadas por humanos
Os humanos fornecem a intenção, o julgamento e a identidade organizacional; os agentes executam.



> *"Desenvolver software nunca foi o objetivo central de uma organização, mas sim o meio para construir os ativos digitais que lhe permitem operar e servir seus clientes. A diferença é que hoje esse meio precisa estar enquadrado dentro da mesma lógica que rege o resto da organização: uma base de conhecimento como fonte de verdade e agentes que operam sobre ela. Quando isso acontece, o desenvolvimento de software deixa de ser uma atividade técnica isolada e se torna mais uma operação da empresa, tão gerenciável, versionável e auditável quanto qualquer outra."*
>
> **Gastón Milano**

Passo 2: Modelar Bases de Conhecimento + Agentes

A arquitetura do software do futuro, segundo Milano, se articula em torno de dois elementos fundamentais: uma **Base de Conhecimento (KB)** que codifica todo o saber da organização, seus processos, políticas, dados, regras de negócio e contexto; e **agentes de IA que operam sobre essa KB** para executar tarefas, tomar decisões e gerar valor.

O modelo redefine os papéis de cada ator na organização:

Papel	Função	Ferramenta
Humanos	Direcionam a intenção, a idealização e a identidade empresarial. Tomam decisões estratégicas.	Base de conhecimento + julgamento humano
IA Generativa	Executa tarefas, gera conteúdo, processa dados.	LLMs + harness de engenharia
Agentes	Operam processos completos de forma autônoma, consomem a KB.	Protocolos agênticos + KB + ferramentas
AI Architect	Projeta a stack, orquestra agentes, define níveis de autonomia.	Orquestradores + plataformas multiagente

> O tempo humano já não é gasto escrevendo código; é gasto definindo o quê, construindo a base de conhecimento e orquestrando os agentes que executam. Gastón Milano ilustra isso com sua própria prática: desde novembro de 2025 ele não escreve uma única linha de código. Seu trabalho é orquestrar.

Passo 3: A economia de tokens

Um dos aspectos práticos que Milano destaca é a necessidade de gerenciar eficientemente o custo dos tokens. As grandes empresas não têm os mesmos subsídios que os usuários individuais; o custo de operar frotas de agentes em escala empresarial pode ser muito significativo e se torna um fator estratégico por si só.

Por essa razão surge o conceito de tokenomics: a economia de tokens. As skills e o conhecimento codificado de maneira precisa permitem reduzir drasticamente a quantidade de tokens consumidos por cada operação, baixando os custos e aumentando a eficiência. Isso transforma a qualidade da base de conhecimento em um fator econômico crítico. Uma KB bem estruturada não apenas faz os agentes trabalharem melhor: faz com que trabalhem mais barato. E em escala, essa diferença define a viabilidade do modelo.

O problema do vibe coding e a necessidade de supervisão

Vibe coding é o termo cunhado por Andrej Karpathy em fevereiro de 2025 para se referir à forma de desenvolver software em que o programador descreve a uma IA o que quer em linguagem natural, aceita o código gerado sem revisá-lo em detalhe, cola os erros diretamente de volta no modelo para que sejam resolvidos, e deixa o sistema crescer de forma orgânica, muitas vezes além do que o próprio desenvolvedor compreende completamente.

Milano descreve esse problema com uma metáfora visual: "O vibe coding pode ser visto como um queijo suíço: cada fatia tem buracos. Quando muitas fatias se combinam, os buracos se acumulam e os erros crescem de forma exponencial. Isso pode funcionar para protótipos rápidos ou projetos pessoais, mas em sistemas empresariais de missão crítica, onde o código precisa ser previsível, sustentável e auditável, o vibe coding sem uma base de conhecimento sólida por trás produz resultados pouco confiáveis."

A solução para esse problema tem várias camadas:

Redução de buracos com melhores ferramentas
A chegada de ferramentas como Claude Code, Codex, Gemini CLI e outros harnesses reduziu significativamente a quantidade de erros na geração automática de código.

Contexto real e preciso
A única forma de reduzir ainda mais os erros é fornecer aos agentes o contexto correto e completo, que vem de uma base de conhecimento bem estruturada.

Supervisão arquitetônica humana
Os humanos devem continuar idealizando as arquiteturas corretas e supervisionando os resultados para manter o software previsível e funcional.

O vibe coding sem contexto é o sintoma. A base de conhecimento é a solução. E o AI Architect é quem une as duas coisas.

AI Architect: o novo papel estratégico

O AI Architect é o papel que emerge quando a geração de código deixa de ser o gargalo. É quem define o quê e o porquê, e quem orquestra os agentes que cuidam do resto.

O surgimento do AI Architect não é casual: é a consequência direta de três fatores que convergiram ao mesmo tempo:

1. Os modelos alcançaram um limiar de competência suficiente para tarefas bem definidas.

2. A engenharia em torno do modelo amadureceu, com ferramentas como Claude Code, Codex e Gemini CLI que resolveram como fazer o loop agêntico, como dar as ferramentas corretas aos agentes, como comprimir a memória e como gerenciar o sistema de arquivos.

3. E os agentes trabalham em paralelo 24 horas por dia, produzindo volumes de código que nenhuma equipe humana consegue igualar.

O resultado é que o gargalo deixou de ser a geração de código para se tornar a definição do que construir e com qual conhecimento fazê-lo. É aí que o AI Architect atua.

Suas responsabilidades centrais são:

- **Projetar o nível de autonomia de cada componente do sistema**
  Determinar quais partes exigem supervisão humana ativa, quais podem operar com sugestões ao humano e quais podem funcionar de maneira totalmente autônoma. Essa decisão, que parece técnica, é na verdade estratégica: define quanto controle a organização retém e onde ela delega aos agentes.

- **Orquestrar frotas de agentes**
  Definir como dezenas ou centenas de agentes trabalhando em paralelo são coordenados, como as tarefas são atribuídas a eles e como seus resultados são gerenciados. Gastón Milano opera habitualmente com 30, 40 ou 50 agentes rodando simultaneamente durante a noite. A orquestração dessa frota é seu trabalho central.

- **Projetar a stack tecnológica completa**
  Escolher quais LLMs usar, quais plataformas de orquestração, quais bancos de dados, como combiná-los. O AI Architect pode combinar Claude, Codex, Gemini ou qualquer outro modelo de acordo com o que cada situação exigir, sem estar preso a um único fornecedor.

- **Garantir a qualidade sem revisão linha a linha**
  É impossível revisar 100.000 linhas de código geradas em um fim de semana por agentes. O AI Architect projeta mecanismos de validação sistêmica e supervisão a posteriori que substituem os fluxos de revisão tradicionais.

- **Manter a visão end to end**
  Preservar a perspectiva integral da solução, sem fragmentá-la artificialmente em dados, UI e segurança como compartimentos estanques que jogam especificações por cima do muro em cada etapa.

Além disso, o AI Architect também define os diferentes níveis de automação que coexistem em um mesmo sistema:

- **Intervenção humana direta**
  Tarefas que ainda exigem que uma pessoa atue sem assistência de agentes: implantações de infraestrutura, configuração de chaves de segurança, instalação de dispositivos IoT, robótica física.

- **Assistência com sugestões**
  O agente propõe, o humano decide. O sistema amplifica a capacidade humana sem substituir o julgamento.

- **Humano como auditor eventual**
  O agente atua de forma autônoma e o humano revisa exceções e resultados, não cada etapa do processo.

- **Autonomia completa**
  O agente opera sem intervenção humana no ciclo normal. Um exemplo concreto: quando a GeneXus gera o código de uma reorganização de banco de dados, ninguém o revisa manualmente. Confia-se que o sistema funciona, porque o contexto e as regras estão corretamente definidos.

Implicações para a Comunidade GeneXus

Para Milano, no mercado atual, todos os usuários da GeneXus são naturalmente "AI Architects". Por quê? Porque o usuário GeneXus já tem incorporado o mindset correto:

- Sabe que primeiro é preciso construir uma base de conhecimento.
- Sabe que é preciso ter os dados corretos antes de construir.
- Entende o conceito de gerar a partir de especificações.
- Pensa de forma end to end sem estar fragmentado em silos tecnológicos.

Além disso, os usuários GeneXus nunca revisaram pull requests nem seguiram fluxos do GitHub. O que em outro contexto poderia parecer uma limitação acaba sendo exatamente a mentalidade correta para este momento, porque esses fluxos de revisão tradicionais estão morrendo.

"Não há como revisar 100.000 linhas de código geradas em um fim de semana por agentes, e toda a indústria está buscando como resolver isso. Os usuários da GeneXus já operavam sem essa dependência. No entanto, há algo que a comunidade ainda precisa incorporar, especificamente o Impact Analysis de dados. Agora existe uma tendência de jogar tudo diretamente no Claude Code sem essa análise prévia, e isso é um erro que sai caro. Incutir essa prática na comunidade é uma tarefa pendente."

Milano antecipa que haverá bases de conhecimento verticais por setor. Isso significa que os parceiros com expertise setorial terão uma oportunidade única de construir ativos de conhecimento profundos e diferenciados para seus setores, que depois se tornarão vantagens competitivas difíceis de replicar.

A GeneXus já está trabalhando para garantir o "future proofing" das bases de conhecimento existentes de seus clientes. O objetivo é que essas KBs possam evoluir para o mundo agêntico sem perder o valor acumulado. A estratégia inclui:

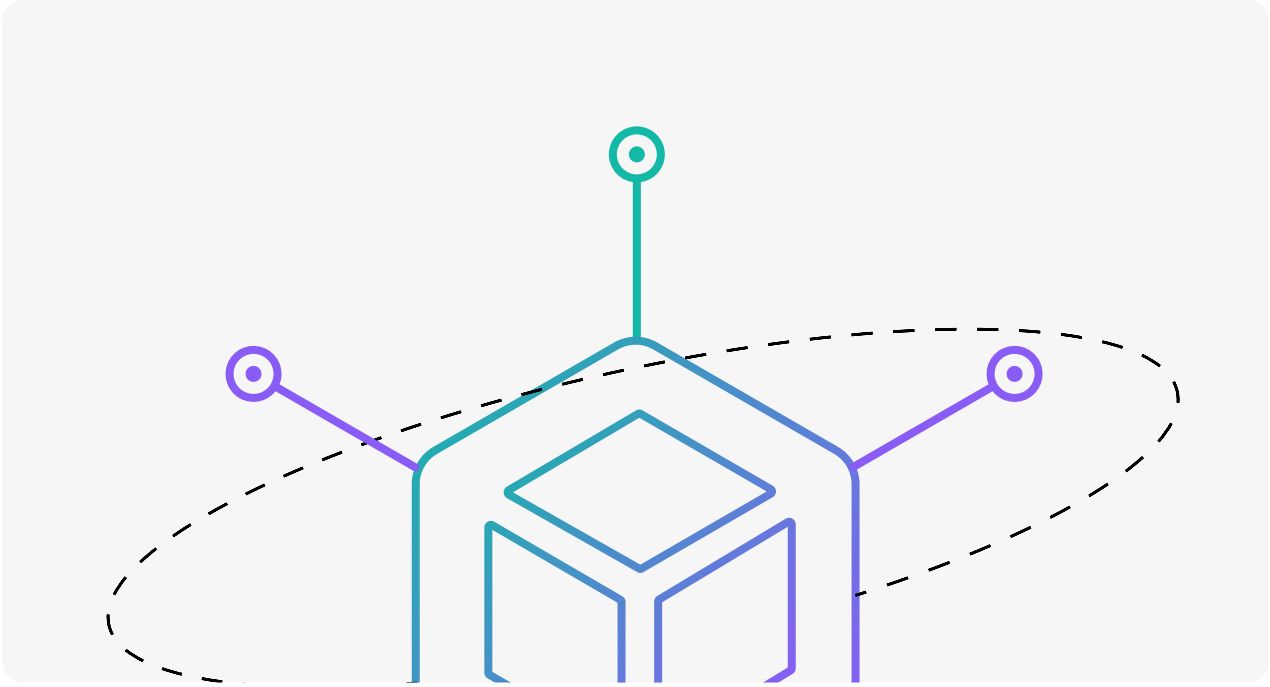
- GeneXus for Agents: ferramenta que torna as KBs consumíveis por agentes externos.
- Segunda iteração rumo à agêntica completa: uma versão evoluída das KBs que permite que os agentes trabalhem diretamente com o conhecimento existente dos clientes.
- Ferramentas de orquestração que permitem combinar a GeneXus com outros geradores e agentes do mercado.

Voltar ao futuro

A GeneXus foi fundada no Uruguai em 1988 com uma ideia que na época parecia radical: que o conhecimento do negócio deveria ser o centro do desenvolvimento de software, e que os sistemas deveriam ser gerados a partir desse conhecimento. Enquanto o resto da indústria construía código, a GeneXus construía bases de conhecimento. Enquanto outros otimizavam como escrever mais rápido, a GeneXus trabalhava para eliminar a escrita manual de código, gerando sistemas a partir do conhecimento declarado.

Hoje, no momento em que a indústria global converge exatamente para o que a GeneXus vem praticando há anos, o modelo evolui mais uma vez: plataforma mais serviços, com bases de conhecimento verticais por setor, implementação, governança e operação contínua. Um modelo recorrente em que o conhecimento codificado é o ativo central e os agentes são o motor de execução.

O que por anos foi uma forma particular de trabalhar acaba sendo hoje a arquitetura correta para a era agêntica.



O momento de agir é agora

O mundo está amadurecendo em uma velocidade extraordinária. Os protocolos agênticos se consolidam. As ferramentas evoluem. E a diferença entre as organizações que já estão construindo suas bases de conhecimento e as que ainda estão avaliando se devem fazê-lo aumenta a cada dia.

As organizações que vão liderar a próxima era do software empresarial são as que entendem, antes das demais, que o ativo estratégico desta era é o conhecimento codificado, e as que têm a orientação correta para construí-lo e operá-lo com agentes.

Esse é exatamente o trabalho que fazemos. Da Globant e da GeneXus, acompanhamos organizações nessa transição: desde a construção de sua primeira base de conhecimento até o design de arquiteturas agênticas completas, a seleção da stack tecnológica e a formação das equipes que vão operar neste novo paradigma.

Se sua organização está buscando dar esse passo, o momento de começar a conversa é agora.

Fale conosco