

AI TRANSLATED · original in German

Bring Your Own Model: GitHub Copilot with Microsoft Foundry, OpenRouter and Local LLMs

Dr. Matthias Liebeck

Azure Cloud Native Cologne · October 13, 2026

◆ INTRO

About me: Matthias Liebeck

- Senior Solution Architect in Düsseldorf, focused on .NET, C# and Azure
- Software developer since 2009, C# since 2006
- 2009 – 2015: Studied computer science & mathematics
- 2018: PhD in computer science on natural language processing / AI
- Organizer of the Azure Düsseldorf Meetup, speaker on Azure & agentic coding
- GitHub Copilot Newsletter: ghcp.liebeck.io



Today's flight plan

- 1 Intro: agentic coding and the harness
- 2 Why there are different models, and who picks them
- 3 Several routes to your own model, including OpenRouter and local
- 4 Microsoft Foundry in detail
- 5 Decision guide and wrap-up

Tonight's question: Who decides which model runs behind your Copilot?

👉 Who knows what agentic coding is?

You describe the goal of a task as a prompt in a chat window. An AI agent plans the steps, reads your code, changes files, runs builds and tests, and checks back when it needs feedback. Not autocomplete, but delegated work.

GitHub Copilot GitHub's coding agent – in the IDE, the CLI and its own app

Claude Code Anthropic's coding agent – terminal, IDE integration and desktop

Codex OpenAI's coding agent – cloud, CLI and IDE extension

👉 Who uses agentic coding?

👉 And with what?

GitHub Copilot

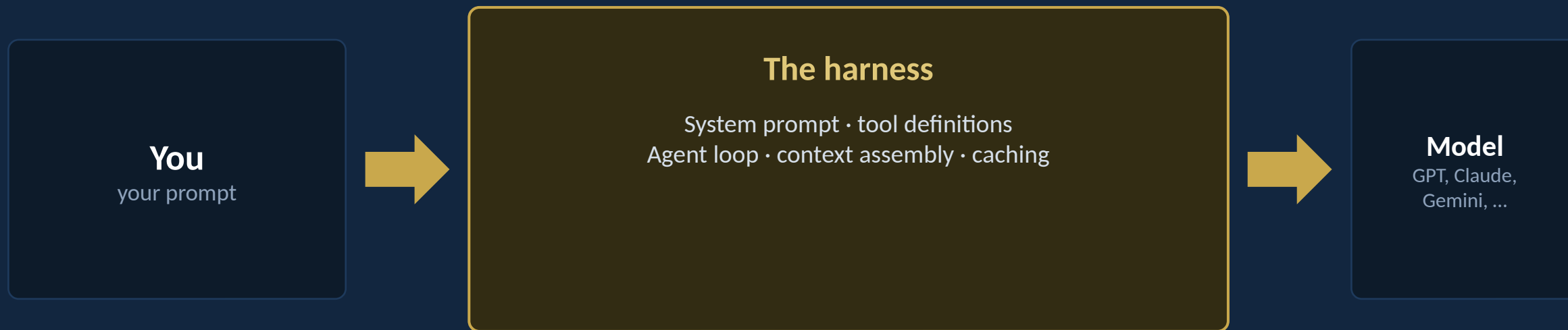
Claude Code

Codex

something else

👉 Who knows what a harness is?

The harness is everything a coding agent builds around the model: system prompt, tools, agent loop and context management. You never talk to the model directly – always to the harness.



We'll see what the harness costs in a moment, with real numbers.



Why are there different models?

Price, capability, and who picks

◆ MODELS

Models differ, in price too

small, fast, cheap

large, thorough, expensive



- A factor of 40 between the cheapest and the most expensive model, for input and output alike
- Large models are usually stronger at planning and hard bugs, small ones are faster
- Context window, provider and data location differ as well
- **But: the cheapest model is not automatically the cheapest coding agent. Every failed attempt costs input again**
- Input / output in \$ per 1M tokens, GitHub Copilot, as of Oct 11, 2026

Which task, which model?

1

Routine: a small model

Commit messages, translations, documentation drafts

2

Planning and hard bugs: a large model

This is where thorough reasoning pays off

3

Review: a different model

Different model family, different blind spots. /rubber-duck does exactly that

4

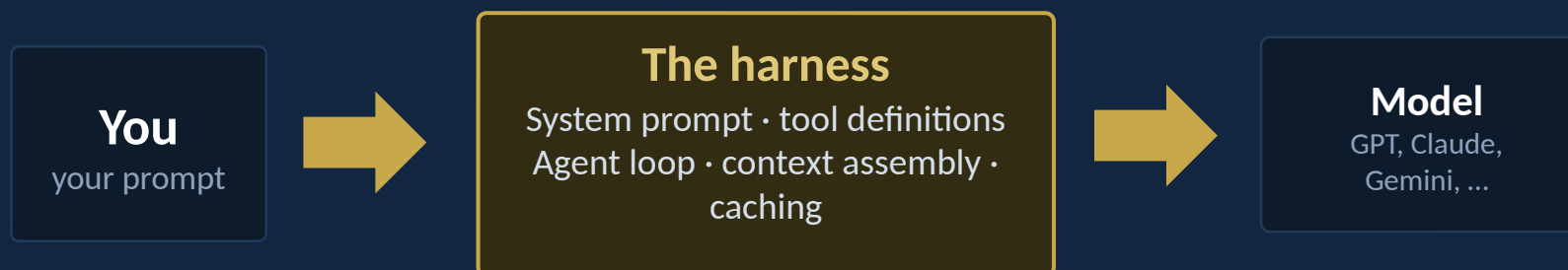
Bulk work: the unit price matters

Many small, uniform tasks

Token types & token-based billing

Token type	What is it?	Why does it matter?
Input	Everything the model reads: harness, context, your prompt	86% of your credits buy input – not output
Output	Everything the model writes: answers, code, tool calls	More expensive per token, but small in volume
Cache Read	Already cached context is re-read cheaply	The cache pays off within a session
Cache Write	Context is written to the cache for reuse	Cache TTL: 300 seconds

- **86% of credits buy input – across all models and tasks (157 runs, 2 production repos)**
- Every new session pays the baseline. Within a session, the cache pays off.



The 20,000-token “OK”

Prompt to Copilot CLI: “Reply only with OK”

~19,752

Input tokens

68

Conversation tokens

12.36

AI Credits

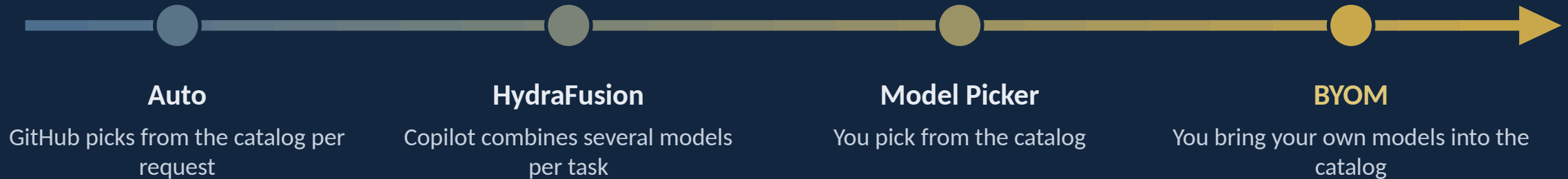
- Practically deterministic: median over 108 runs: 21,103 tokens (19,787–23,328)
- The actual conversation makes up 0.3% of the input tokens
- Measured: Copilot CLI 1.0.75, gpt-5.6-sol, session.shutdown events (events.jsonl)

◆ MODEL CHOICE

Who decides which model runs?

no setup, GitHub decides

your own contract, you decide



- All four run in the same harness. The difference is who picks and where the model comes from
- BYOK, Bring Your Own Key: your own access to the provider
- BYOM, Bring Your Own Model: the model you connect with it
- From here on, we talk about BYOM

Auto with tiers: GitHub picks

Efficiency

Low cost, for quick, simple tasks

Balance

Cost, quality and latency balanced, for everyday work

Intelligence

Quality first, for complex tasks

- **All three tiers use the same models. Auto evaluates each prompt individually and picks the right model**
- Example: a docstring can get a small model even on Intelligence
- You are billed for the chosen model, regardless of tier. Paid plans get a 10% discount
- Rolling out in VS Code, Copilot CLI and the app. According to GitHub, the first step toward more influence on model choice
- Convenient, but GitHub decides. No contract of your own, no region of your own

HydraFusion: several models per task

Single

One chosen model solves the task directly

Cascade

An efficient model produces a draft. A quality gate accepts it or escalates to a stronger model

Critique

One model drafts. A critic from another model family reviews read-only, like Rubber Duck. Then it is revised once

- Choosing the workflow is an optimization problem: signals for reasoning, code generation, debugging and tool use determine the cheapest workflow that meets the quality bar
- Auto picks a model per request, HydraFusion picks a workflow
- You pay for the tokens of the models used at the normal price, with no surcharge
- Research preview: Copilot CLI since Sep 4, app and VS Code since Sep 30

HydraFusion: numbers and your own models

GitHub figures, baseline Claude Opus 5	Cost	Quality
TerminalBench 2.1	-67%	+4.9 points
DeepSWE	-36%	-1.5 points
CheckpointBench (internal)	-65%	-0.1 points

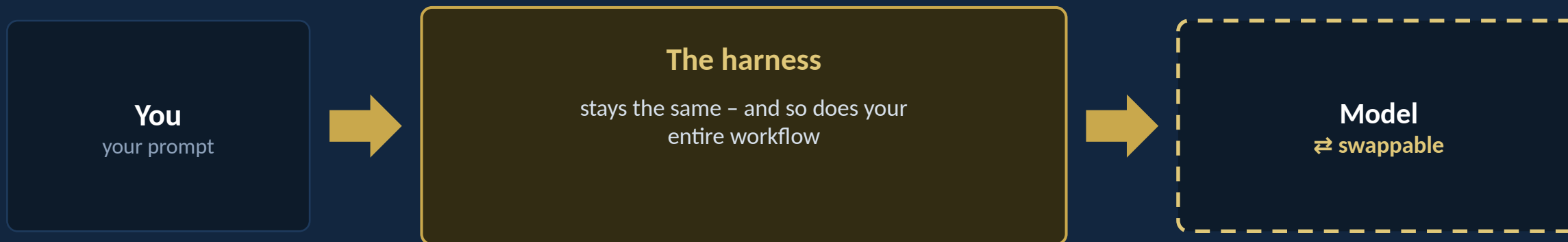
- Much cheaper, quality roughly at Opus level
- Measured on the same benchmarks HydraFusion was tuned on

And your own models?

- HydraFusion picks from a fixed pool of models
- Lerna, a community plugin for Copilot CLI, lists six model IDs: gpt-5.6-sol, -luna, -terra, claude-opus-5 and two MAI-Code variants
- Lerna redirects the four GPT and Claude models to your own Foundry deployments. The MAI models stay with Copilot
- As of Oct 11: HydraFusion works only partially with BYOM

◆ MODEL CHOICE

Why your own key or model at all?



The block on the right is swappable. The one in the middle stays – you learn nothing new, you only swap the model.

Billing

Your own contract, your own cost control – billed directly by the provider

Use your existing GPU

Local models without the cloud – if memory allows

Compliance & location

Not allowed everywhere: some teams may only use on-premises or Europe

And as we'll see next: there are several routes to get there.



Several routes to your own model

The most common ones at a glance

The most common routes to your model

1 • Copilot models

Default: GitHub picks and hosts – billed via Copilot credits

2 • Provider direct

Your own API key (Anthropic, OpenAI, ...) – billed by the provider, doesn't count against the Copilot quota

3 • OpenRouter

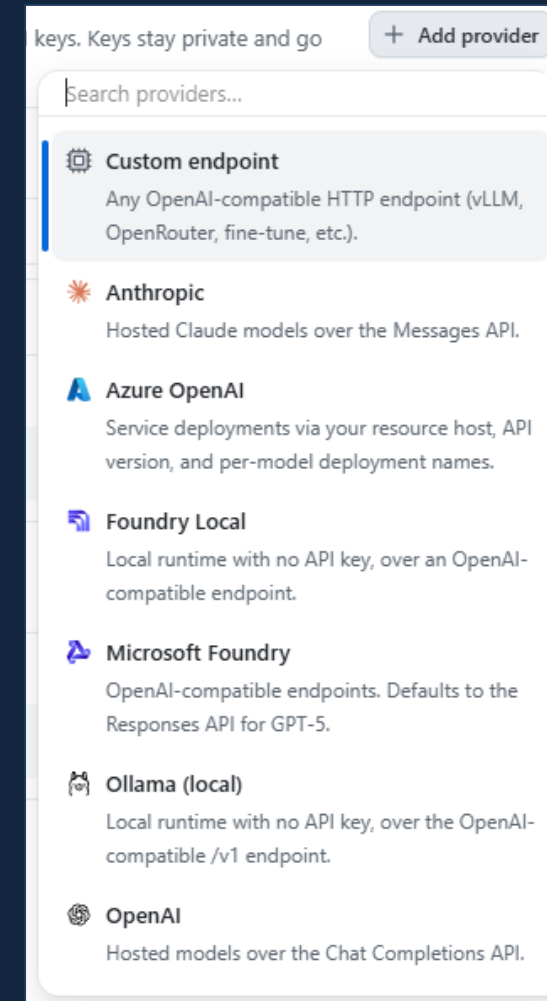
One key, hundreds of models – including large open-weight models, hosted remotely

4 • Local (Ollama)

Full control, data stays local – the hardware decides

5 • Microsoft Foundry

Your own deployment in your own tenant – region selectable, billed as Azure consumption



Add Provider in the Copilot app

Two ways to connect your own models

Personal BYOK

- You enter key and model yourself in the client
- The key stays on your machine
- The request goes straight to the provider, without the Copilot API
- Works without a Copilot license
- VS Code, JetBrains, Xcode, CLI, App, SDK
- Admins can disable it via policy

Enterprise BYOK

- Enterprise or org owners store keys centrally
- The models appear in the model picker under the company name
- The request goes through GitHub's Copilot API
- Requires Copilot Business or Enterprise
- Copilot Chat, CLI and IDEs
- Public preview

Where your own models work today

Surface	Personal BYOK	Enterprise BYOK	Tested myself
Copilot App	Yes, preview	Yes	Yes
Copilot CLI	Yes	Yes	Yes
Copilot SDK	Yes	Yes	Yes
VS Code, JetBrains, Xcode	Yes	Yes	No
Visual Studio 18.10	New agent, preview	Yes	No
Copilot Chat on github.com	No	Yes	No

Sources: GitHub docs, Visual Studio release notes, as of October 2026

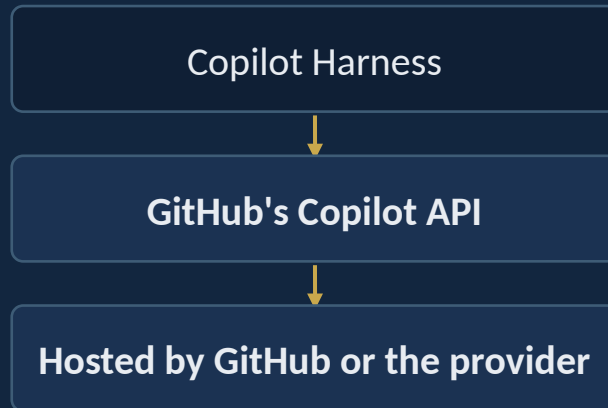
What BYOM doesn't cover

- Inline completions keep running on Copilot's own models
- Indexing and code search can still run through Copilot. Your model receives the matches as context
- Enterprise BYOK: the Copilot license stays, and you pay for tokens on top at the provider
- Filters and privacy depend on whether the request goes through the Copilot API or directly to the provider
- In general: much of this is public preview and changes weekly

BYOM replaces the model, not the product.

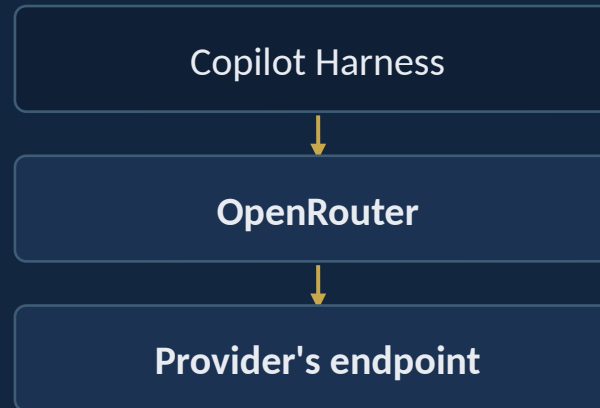
Where do my prompts go?

GitHub model



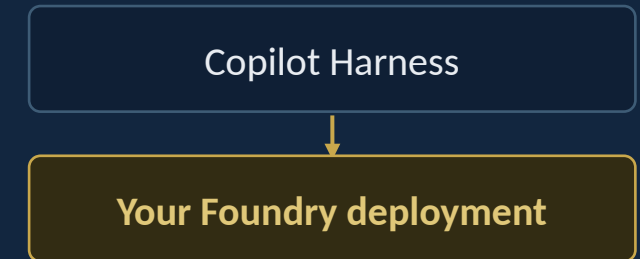
Sees the prompt: GitHub and possibly the model provider

OpenRouter



*Sees the prompt: OpenRouter and the provider.
Retention depends on settings*

Microsoft Foundry



Sees the prompt: Microsoft, location depends on deployment type

- OpenRouter and Foundry shown here with personal BYOK: straight to the provider. Enterprise BYOK also goes through the Copilot API
- Resource location is not processing location: my Luna deployment sits in Sweden Central, but as Global Standard it can be processed worldwide
- Only the inference path is shown. Telemetry, indexing and MCP tools have their own paths

OpenRouter: hundreds of models, one key

- An aggregator, not a model provider: routes requests to the providers behind it. Cost per request on the Activity page
- Prepaid credit · list prices plus a 5.5% platform fee on top-up (pay as you go) · empty balance stops the calls
- The real win: large open-weight models (DeepSeek, Qwen, Kimi) without your own GPU
- Add it in the Copilot app as a custom endpoint (OpenAI-compatible). Tested myself with qwen3-coder-next: tool calling was stable

The screenshot displays the OpenRouter website interface. On the left is a sidebar with navigation options: Input modalities (Text, Image, File, Audio, Video), Discounted, Context length, Prompt pricing, Series, Categories, Supported parameters, Distillable, Zero data retention, and In-region routing. The main content area is titled 'Models' and features a search bar, sorting options (Newest, All, Pinned, List, Table), and a filter bar showing counts for various modalities: Text (430), Image (50), Video (28), Speech (18), Transcription (20), Embeddings (37), Rerank (7), and Audio (4). Three model cards are visible, all for 'OpenAI: GPT-6 Astra' variants. The first card is 'GPT-6 Astra (batch)' with 64.3M tokens, a description of its use cases, and pricing of \$5/M input and \$25/M output tokens. The second card is 'GPT-6 Astra' with 179B tokens, categorized for Programming, Technology, and Trivia, with pricing of \$10/M input and \$50/M output tokens. The third card is 'GPT-6 Astra Pro (batch)' with 11.9M tokens, noting it uses a reasoning mode for higher quality, with pricing of \$5/M input and \$25/M output tokens.

One model, many providers

Kimi K3, \$ per 1M tokens	Input	Output
Cheapest input (Wafer)	0.33	14.89
Cheapest output (Decart, discounted)	1.98	9.90
List price Moonshot and GitHub Copilot	3.00	15.00
Most expensive provider (Fireworks Fast)	4.50	22.50

- Same model: input \$0.33 to \$4.50, output \$9.90 to \$22.50 depending on the endpoint
- The cheapest input is not the cheapest output. 86% of credits buy input, so the input price matters most
- Providers also differ in quantization, speed, context window and reliability. Pin the provider
- OpenRouter model page, as of Oct 11, 2026

OpenRouter: Who processes my code?

Risks

- An additional service between Copilot and the provider
- The same model at different providers, with different privacy rules
- Routing and fallbacks can switch the provider. Happened to me: ordered Qwen, Novita delivered
- No training, no retention and EU processing are three different requirements

The switches against it

- Zero Data Retention: only endpoints whose ZDR policy provides for no persistent storage of prompts and responses
- Data Collection Deny: exclude providers that collect data or train on it
- Provider allowlist: approved providers only, fallbacks off
- Region: restrict processing where account and provider support it

From Copilot, you set these switches in the OpenRouter settings, not per request

A model name is no privacy guarantee. What counts is the endpoint that answers.

Local models

Local on my hardware (RTX 3050, 8 GB VRAM):

- The mechanics work: the Ollama model shows up in the picker and answers
- My test: with quantized 8B models, agentic coding was not reliable enough, especially for tool calling
- Windows pitfall: set the context window via the UI slider, otherwise output is silently cut off mid-word

New since October 7, not tested myself:

- Copilot CLI: /model finds running Ollama models, switch without restart
- Requirement: the model supports tool calling and streaming
- A local model doesn't mean Copilot works offline. Check telemetry and other network access separately

TAKEAWAY

Local models work technically. For coding agents, my hardware is still the limit.



Route 5: Microsoft Foundry

- **DeepSeek-V4-Pro as a serverless deployment in my own tenant, endpoint plus key in the app**
- For my test: serverless API with pay-per-token. Managed compute means dedicated GPU capacity, billed by time
- At the time of my test, Qwen3-Coder-Next was only available via managed compute
- Enterprise angle: deployment and billing in your own tenant. The deployment type decides where processing happens
- **Tested: a real feature on my own repo, stable tool calling, good clarifying questions**

- ▼ **Availability**
 - ☐ All models
 - ☒ Available in my project ⓘ
- Collections
- Source
- Supported features
- Inference tasks
- ▼ **Deployment options (1)**
 - ☐ Managed compute 96
 - ☒ Serverless API 98
- Deployment SKU

Foundry catalog: Serverless API

What does BYOM really cost?

GPT-5.6 Luna, \$ per 1M	Input / Output	Surcharge	Where to see costs
GitHub Copilot	0.20 / 1.20	none	GitHub Usage
OpenRouter	0.20 / 1.20	5.5% on top-up	Activity page
Azure EU as a provider on OpenRouter	0.22 / 1.32	Data Zone EU approx. +10%	Activity page
Microsoft Foundry	per deployment type, pricing calculator	Global, Data Zone, Regional	Azure Cost Management

- List prices are only partly comparable: Copilot allowances, token billing and top-up fees differ
- GitHub and OpenRouter model page, as of Oct 11, 2026

The token price isn't the only difference. Cache, deployment type and fees count too.

Models come and go

- Sep 10: GitHub deprecates MAI-Code-1-Flash
- Oct 2: Kimi K2.7 Code, Claude Opus 4.7, Gemini 3.5 and 3.6 Flash follow
- With your own deployment, you decide when to switch
- Foundry removes models too, e.g. DeepSeek-V3.1 from the Model Router

◆ ROUTE 5 • FOUNDRY

When is your own tenant worth it?

1

Deployment control: Prompts go to your own deployment. The deployment type decides where processing happens: Global, Data Zone or Regional.

2

Pick the version deliberately: Subscription models can change on the server side. Your own deployment stays stable within its lifecycle window. If you measure and compare, that's exactly what you need.

3

Cost model: pay per token on the Azure bill instead of AI credits: your own contract, your own cost control.

4

Model choice: the Foundry catalog goes beyond the subscription models.

BYOM is not an automatic quality upgrade, but more control over model choice, operations and cost.

Foundry in the enterprise

Azure governance

- Your own subscription, central billing
- Choose the deployment type deliberately: Global, Data Zone or Regional
- Budgets and alerts. They don't stop requests
- Check content filters per deployment
- For partner models, check carefully who hosts them

GitHub governance

- Enterprise BYOK: store keys centrally
- Enable models per organization
- Copilot Business or Enterprise is still required
- The Copilot API stays in the inference path
- Public preview, not tested myself



Wrap-up

◆ DECISION GUIDE

Which route fits me?

No setup, no second contract?



Route 1: Copilot models, ideally with Auto

Existing contract with Anthropic or OpenAI?
(and plenty of money for tokens 😄)



Route 2: provider direct

Want to try out many models?



Route 3: OpenRouter

Lots of GPU memory, data must not leave?



Route 4: local

Already on Azure, central governance or data residency requirements?



Route 5: Foundry, check the deployment type

Your first 30 minutes with Foundry

- 1 Create a budget with an alert**
Warns at thresholds, but doesn't stop requests
- 2 Deploy a model serverless**
A model with tool calling; choose the deployment type deliberately: Global, Data Zone or Regional
- 3 Add the provider in the Copilot app**
The inference endpoint, not the project endpoint
- 4 One small real task**
A non-critical repo, then check tool calling, token usage and cost

Cost Management shows costs with a delay of a few hours. That's fine for a first test.

Three sentences to take home

1 Copilot provides the harness.

Agent loop, tools and workflow stay the same, whichever model runs.

2 BYOM gives you the choice of model.

Models differ in price by a factor of 40. The cheapest model is not automatically the cheapest agent.

3 Foundry brings deployment and billing into your Azure tenant.

The deployment type decides where processing happens.



Thank you



Connect on LinkedIn

linkedin.com/in/matthias-liebeck



GitHub Copilot Newsletter

ghcp.liebeck.io