# THE OPERATING SYSTEM IS BECOMING AGENTIC

*From AI Assistants to Agentic Systems*

Written by: Jorge Pereira

With the help from my research assistants/ editors:
OpenAI ChatGPT, Anthropic Claude, Gemini and Perplexity.

Published August 30, 2026

Original article: https://jorgep.com/blog/the-operating-system-is-becoming-agentic/


## About Jorge:

AI Adoption & Technology Strategy | Helping people and organizations turn AI, automation, and emerging technologies into practical business outcomes | Workplace Transformation

Contact Jorge at: https://jorgep.com/about-jorge/

## From AI Assistants to Agentic Systems

Something interesting has been happening in AI over the past four to six weeks that I think is bigger than the arrival of another chatbot, another model, or even another generation of AI agents.

I've been spending time experimenting with platforms such as Buzz, Omarchy OS, the latest Microsoft Copilot platform changes, newer versions of Hermes Desktop, and several other emerging agent environments. Individually, each one is interesting. But taken together, they point to a much larger trend.

**The shift I see is not simply toward smarter assistants.**

It is toward a new way of interacting with computers—one where agents increasingly become part of the operating environment itself, work across applications and systems, collaborate with other agents, and involve the human primarily for direction, judgment, and approval.

In other words, I think we are starting to see the early shape of something much bigger:

**The operating system is becoming agentic.**

# The way we interact with computers is beginning to change.

For most of the history of personal computing, we have adapted ourselves to the computer.

First, we learned commands.

Then came windows, menus, icons, folders, applications, browser tabs, and mobile apps.

More recently, we learned how to prompt AI assistants.

But I think we are approaching another fundamental transition.

> *What happens when we stop operating the computer—and start directing it?*

For the last few years, most of us have interacted with AI through an assistant.

We open ChatGPT, Claude, Gemini, an AI-enabled application, or one of the growing number of personal-agent frameworks. We tell it what we want. It performs some work. We review the result. Then we decide what happens next.

Systems such as ChatGPT Work, Claude's agentic tools, Hermes Agent, Agent Zero, OpenClaw, and the rapidly expanding family of OpenClaw-derived projects are already pushing well beyond simple question-and-answer interaction.

- They can use tools.
- They can browse.
- They can write code.
- They can interact with files.
- They can maintain context.
- Some can schedule work, create subagents, operate remote computers, or continue executing after the initial prompt.

These are significant advances. But much of our thinking is still based on a familiar model:

***There is a human on one side and an AI assistant on the other.***

I increasingly think that is only an intermediate stage.

The next phase isn't simply a more capable AI assistant.

**It is an agentic computing environment.**

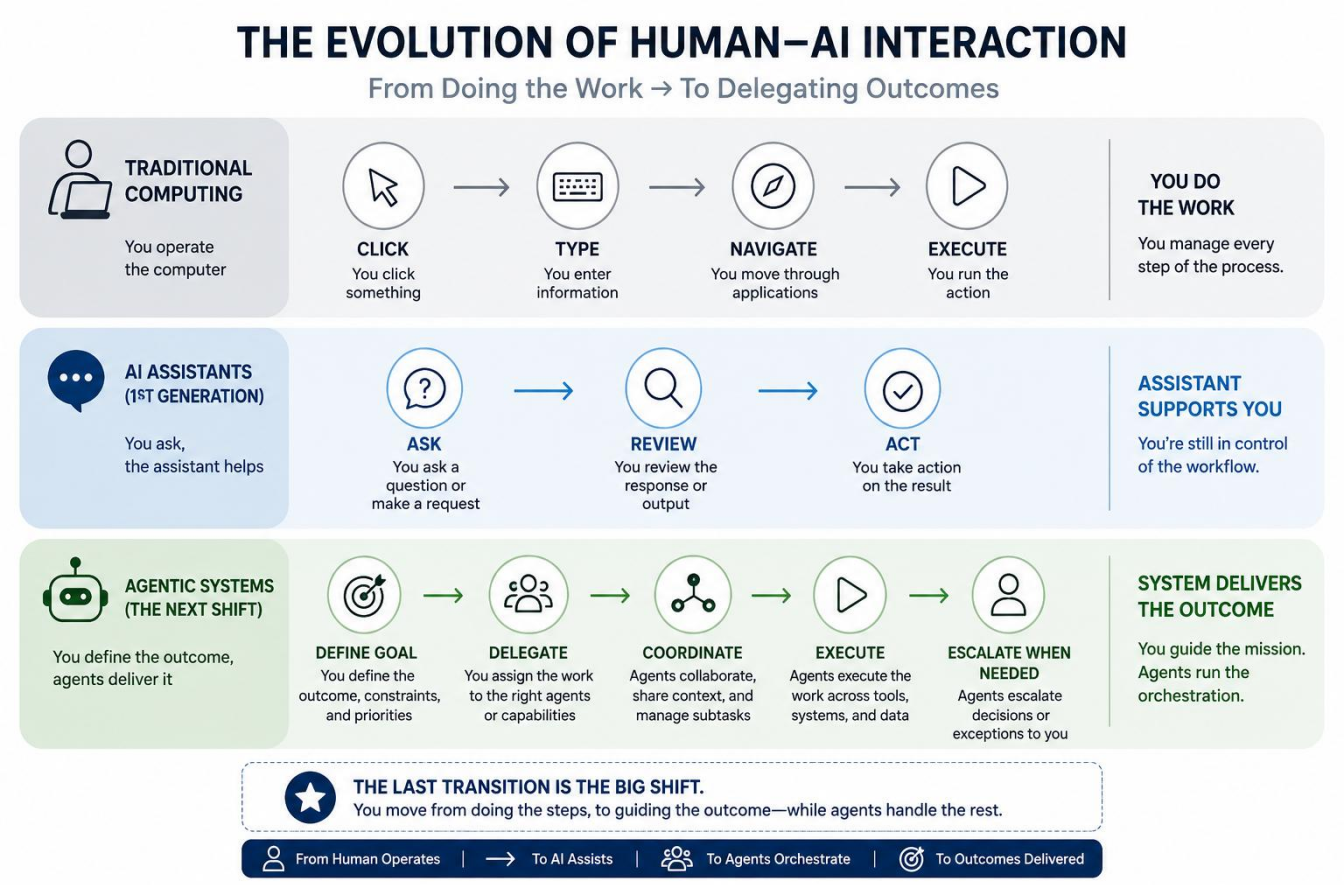
And that could fundamentally change the relationship between humans, applications, operating systems, networks, and computers.

## From AI Assistants to Agentic Systems

The evolution from AI assistants to agentic systems is not simply about giving AI more capabilities. It changes who is responsible for orchestrating the work.

With traditional computing, the human performs every step. With AI assistants, the human still directs the workflow while the AI helps with individual tasks. Agentic systems begin to move that orchestration into the system itself—allowing agents to coordinate, delegate, execute, and bring the human back in when judgment or approval is needed.

The graphic below illustrates that progression.



**That last transition is the important one.**

Today, the human is still frequently the orchestration engine. We decide which application to open, move information between systems, choose which AI tool should handle the next step, and initiate each stage of the workflow.

In an agentic environment, the human begins to move up a level—from managing individual steps to defining the goal, setting the boundaries, and guiding the outcome.

Instead of asking an individual AI assistant to perform a specific task, we begin giving an environment an objective and allowing the environment to determine which agents, tools, computers, models, and applications should participate.

| AI Assistant Model | Agentic Systems Model |
|---|---|
| Human selects the application | Human defines the objective |
| Human selects the AI | System selects appropriate capabilities |
| Human moves context between tools | Agents exchange context |
| Human initiates most steps | Agents delegate subtasks |
| AI waits for another prompt | Agents can continue working |
| Human manages every boundary | Humans manage decisions and exceptions |
| One primary execution environment | Local + private + cloud environments |
| One assistant | Multiple cooperating agents |

The shift is subtle at first.

**But architecturally, it is enormous.**

## The Operating System Becomes an Active Participant

Historically, the operating system has been relatively passive from the user's perspective.

It manages:

- hardware
- memory
- processes
- networking
- storage
- applications
- security
- and the graphical interface

But the human still decides what happens.

- You launch the application.
- You choose the file.
- You navigate the interface.
- You initiate the process.
- You move the information.

An agentic operating environment begins to change that relationship.

The operating system can increasingly become a place where agents perceive, reason, act, coordinate, and continue working.

Instead of simply hosting applications, it starts participating in the orchestration of work.

That is what I mean when I say:

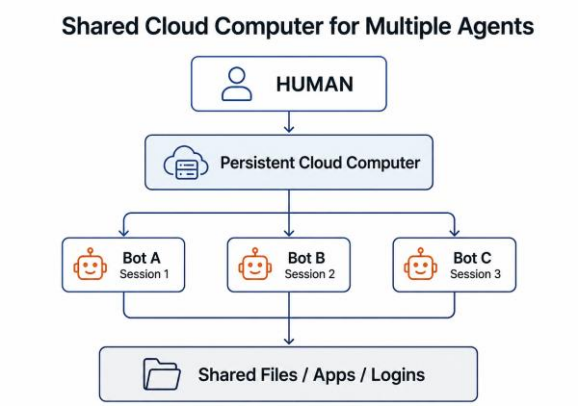
**The operating system is becoming agentic.**

## Grok Bot: Give the Agent Its Own Environment

One of the clearest examples of this emerging direction is Grok Bot.

Rather than thinking only in terms of an AI conversation, the model moves toward agents working inside a persistent cloud computing environment.

With Grok Bot, the user gets a persistent cloud computer that can be used by multiple Bots. Each Bot operates through its own screen or session, allowing several agents to work in parallel while sharing access to the same files, browser sessions, applications, and authenticated services.




That distinction is important.

**It is not necessarily one computer per agent.**

It is more like giving a team of agents access to a shared digital workplace.

Conceptually, it looks something like this:

Now think about the difference in how work is assigned.

An AI assistant might receive:

> *Help me write a follow-up email to these customers.*

An agent operating inside this type of environment might receive:

> *Follow up with yesterday's customers, update the appropriate records, and come back to me when something requires my decision.*

Those are fundamentally different interactions.

The first produces something for the human to execute.

The second begins executing the workflow itself.

The agent can open applications, use a browser, move between systems, enter information, and continue working without requiring the human to manually drive every step.

And because multiple agents can operate in parallel, the system begins to look less like a single assistant and more like a digital team sharing a computing environment.

That is a significant architectural shift.

The innovation is not simply a smarter model.

**intelligence + tools + identity + permissions + persistent computing environment + parallel agents**

The agents do not just have access to software.

**They have somewhere to work.**

And increasingly, that working environment can exist independently of the computer the human happens to be using.

That distinction matters.

The computer is no longer only an interface for the human.

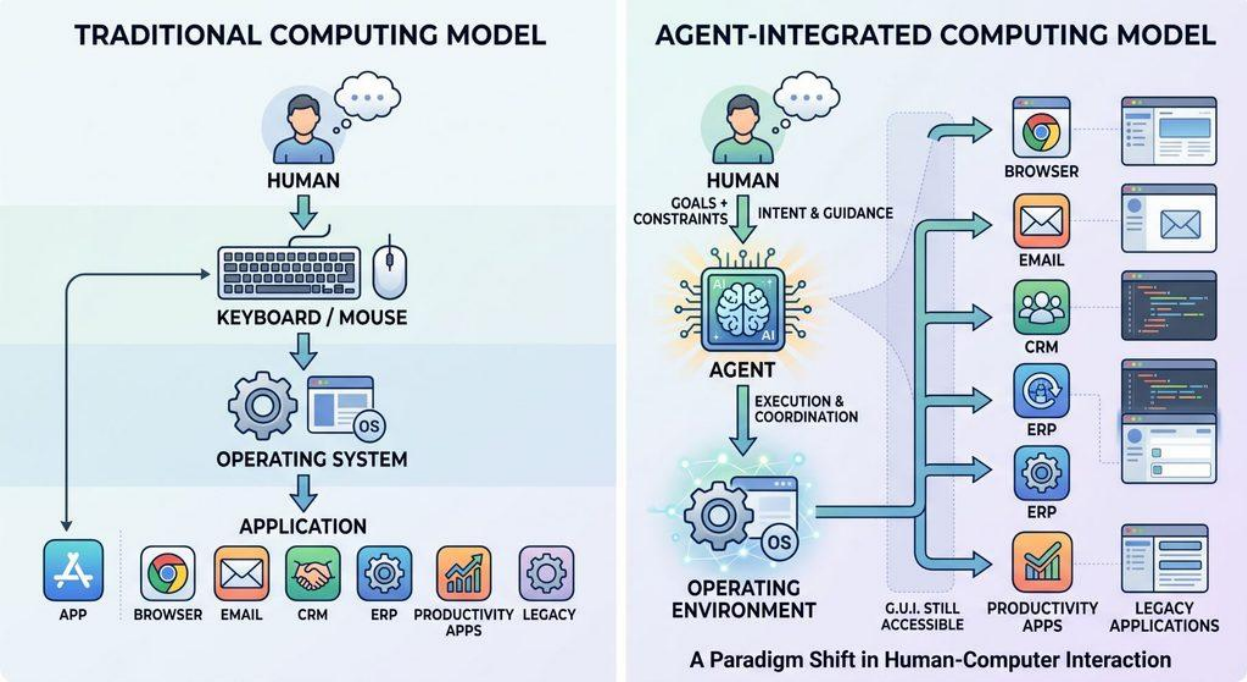
**It is becoming a workspace that agents can use directly.**

## Warmwind OS: What If the Agent Operates the Computer?

Warmwind OS approaches the problem even more directly.

Instead of thinking about AI simply as another application running on a computer, Warmwind explores what happens when the computing environment itself is designed around AI workers.

- Agents can operate cloud-hosted computers.
- They can see graphical interfaces.
- They can click.
- Type.
- Navigate.
- Use browsers.
- Interact with applications.
- And potentially continue working even when the human is no longer sitting in front of the computer.




The graphical interface does not disappear.

**But the human may no longer be the only one operating it.**

That is a very different way to think about the GUI.

For decades, we have designed graphical interfaces entirely around people.

Now we may increasingly be designing systems where both humans and agents interact with software.

### This also begs the question –

**Why do we need the GUI if we can naturally talk to it?** - More on this later!

## APIs Are No Longer the Only Door

This could also have enormous implications for enterprise automation.

Traditional automation depends heavily on integration.

If an application provides a good API, automation can be relatively straightforward.

If it doesn't? Things become much more difficult.

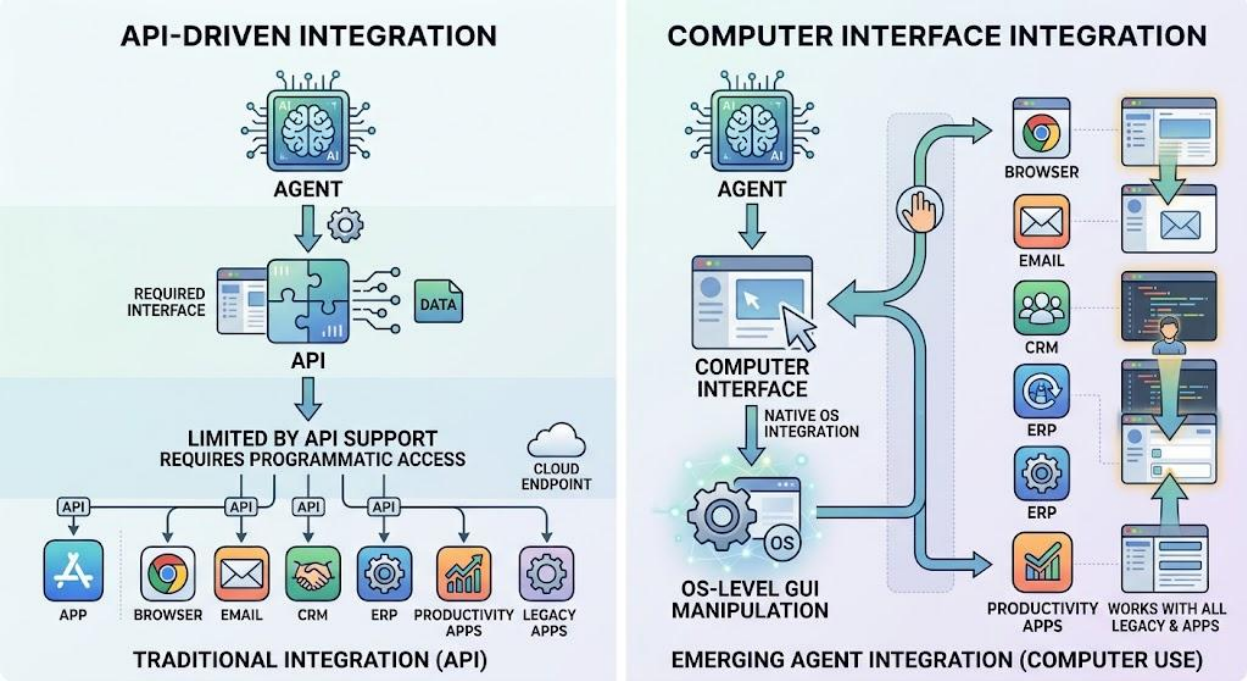
That has produced decades of:

- APIs.
- Connectors.
- Middleware.
- RPA.
- ETL.

- Scripts.
- Custom integrations.
- Workflow platforms.

All of those approaches will continue to matter.

But agents capable of computer use introduce another possibility.




Suddenly, an internal application written twenty years ago may potentially participate in an agentic workflow even if nobody ever built a modern API for it.

That doesn't mean GUI automation replaces APIs.

Far from it.

APIs are generally faster, more predictable, structured, and easier to govern.

But visual computer use gives agents another option.

And I think that becomes one of the defining characteristics of agentic computing.

**Agents will use whichever interaction method makes the most sense.**

| Need | Possible Interaction |
|---|---|
| Structured application | API |
| Tools and enterprise data | MCP |
| Another autonomous agent | A2A |
| Human judgment | A2H |

| | |
|---|---|
| Legacy application | Computer use / GUI |
| Local files | Direct local access |
| Private database | Private-network agent |
| Physical device | Local/edge agent |
| Cloud application | API, browser, or cloud computer |

That is why I don't believe the future will be defined by a single agent architecture.

**It will be hybrid.**

## Omarchy: Agents Become Operating-System Citizens

Another interesting example is Omarchy.

It approaches the idea from a somewhat different direction.

Rather than creating an entirely separate cloud environment for agents, Omarchy increasingly treats multiple AI coding agents and AI tools as normal participants in the operating system.

The important idea isn't which particular coding agent someone chooses.

It is that choosing and invoking an agent begins to feel more like choosing a native system capability.

Today we still frequently think:

> *I'm going to launch an AI application.*

But the emerging model is closer to:

> *AI capabilities are simply part of my computing environment.*

That might sound like a small distinction.

I don't think it is.

Consider networking.

There was a time when going "online" was a deliberate activity.

You launched software. Established a connection. Connected to a remote system.

Eventually networking disappeared into the infrastructure of computing itself.

We stopped thinking about launching the Internet.

Connectivity simply became part of the operating environment.

AI could follow a similar path.

And this is what the operating system becoming agentic begins to look like.

Agents are no longer simply destinations we visit.

**They increasingly become first-class capabilities of the environment itself.**

## And Then the Agents Start Talking to Each Other

Once we have multiple agents, another question immediately appears:

**How do they work together?**

Imagine a future where you have:

- A research agent.
- A finance agent.
- A coding agent.
- A local private agent.
- A CRM agent.
- A customer service agent.
- A cloud computer agent.
- A personal communications agent.
- A specialized industry agent.

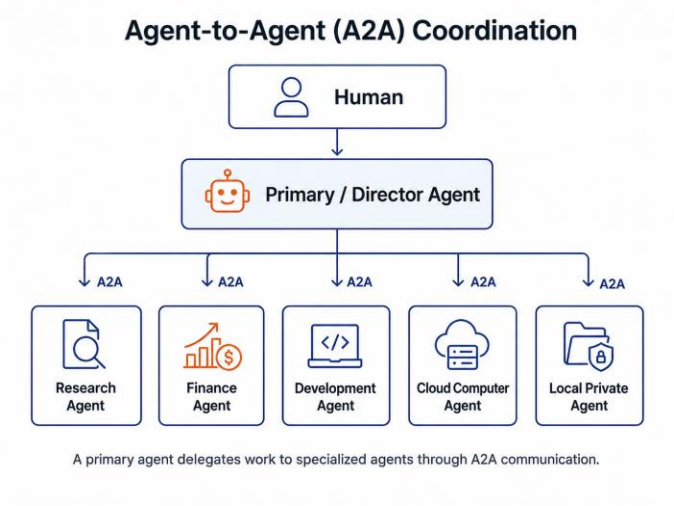
One giant general-purpose agent could theoretically attempt to do everything. But that may not be the best architecture.

The same reason we have specialized people, applications, services, and systems applies to agents.

Different agents will have different:

- capabilities
- permissions
- models
- tools
- locations
- costs
- data
- security boundaries
- and areas of expertise

**That is where Agent-to-Agent communication—A2A—becomes important.**




Now we are no longer talking about an assistant.

**We are talking about a network of capabilities.**

Agents need ways to:

- Discover one another.
- Advertise what they can do.
- Request work.
- Delegate subtasks.
- Transfer state.
- Exchange results.
- Communicate status.
- Negotiate capabilities.

That is why emerging standards such as A2A are potentially so important.

The future of agents may depend just as much on interoperability as it does on intelligence.

## A2A Is Only Half of the Equation

There is another relationship that may be even more important.

**Agent-to-Human—A2H.**

If agents become increasingly autonomous, humans should not have to approve every tiny operation.

That would defeat much of the purpose.

But we also don't want agents independently making every consequential decision.

**The better model is appropriate human involvement.**

An agent operates within its established boundaries and comes back to a human when it encounters something requiring judgment, permission, policy interpretation, or risk acceptance.

For example, imagine an agent arranging business travel.

- Check your calendar.
- Research flights.
- Compare fares.
- Evaluate hotels.
- Check travel time.
- Review corporate travel policy.
- Coordinate the itinerary.

Then it discovers:

*The best flight exceeds company policy by $375 but saves four hours of travel time. Approve?*

That is an excellent A2H interaction.

The human doesn't need to navigate twelve websites and compare twenty flights.

The agent handles the mechanics.

**The human handles the judgment.**

That is a very important distinction.

## MCP + A2A + A2H Starts to Look Like an Agent Fabric

Now put these emerging pieces together.

| Layer | Role |
|---|---|
| LLMs | Reasoning, language, planning |
| MCP | Agent ↔ tools and data |
| A2A | Agent ↔ agent |
| A2H | Agent ↔ human |
| Computer Use | Agent ↔ graphical software |
| Identity | Who or what the agent is |
| Permissions | What the agent may access |
| Policy | What the agent may do |
| Orchestration | Which capability should perform the work |
| Observability | What happened and why |
| Operating Environment | Where the agent executes |

Individually, each is interesting.

Together, they start to look like something much larger.

**An agent fabric.**

A human could provide an objective to one agent.

That agent might access tools through MCP.

Delegate research to another agent through A2A.

Ask a private local agent to process confidential information.

Launch a cloud computer agent to interact with an application.

Then escalate an important decision back to the human using an A2H interaction.

That is not simply an AI assistant.

**That is a distributed computing environment orchestrated around intent.**

## Where Platforms Like Buzz Fit

We are also beginning to see agentic systems emerge around specific business outcomes.

Buzz.ai, for example, applies agentic concepts to sales workflows.

Instead of simply asking AI to write an outbound email, systems in this category can increasingly participate across more of the workflow:

- Identify prospects.
- Research companies.
- Enrich contact information.
- Develop messaging.
- Execute outreach.
- React to engagement.
- Update business systems.
- Escalate opportunities.

Buzz itself shouldn't necessarily be thought of as a general-purpose A2A operating fabric.

But it illustrates an important transition.

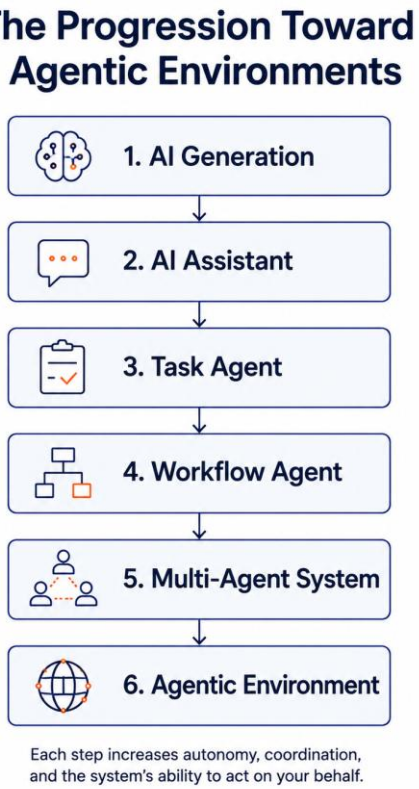
AI is moving from:

> *Generate something for me.*

toward:

> *Take responsibility for this outcome.*

The progression looks something like this:




Each evolutionary stage has steadily abstracted the human further away from micro-execution:

- AI Generation: Isolated, stateless text and image completions triggered by direct single prompts.
- AI Assistant: Conversational sidecars (chat windows, copilots) retaining basic session context and answering queries.
- Task Agent: Single-purpose tools capable of executing a bounded action (for example, scraping a page or writing a code snippet).
- Workflow Agent: Multi-step executors that plan, handle tool calling, and manage state across an entire pipeline.
- Multi-Agent System: Specialized autonomous units communicating and delegating sub-tasks to one another.
- Agentic Environment: An ambient operating layer where hybrid local and cloud agents can coordinate around outcomes.

And the boundaries between those categories are already becoming blurry.

## The OpenClaw Effect

I also think the OpenClaw ecosystem is important to watch.

OpenClaw and what I jokingly think of as the growing collection of “OpenClaw babies” demonstrate another side of the agentic transition.

You don’t necessarily need a giant cloud platform to participate.

Powerful personal agents can increasingly live on machines we control.

- They can use local or hosted models.
- Access local files.
- Maintain persistent state.
- Interact through messaging platforms.
- Use tools.
- Run commands.
- Participate in workflows.
- And connect to other services.

This matters because the future probably isn’t one enormous AI agent that controls everything.

It may be thousands—or eventually millions—of different agents.

- Some corporate.
- Some personal.
- Some local.
- Some cloud-based.
- Some highly specialized.
- Some temporary.
- Some persistent.

Which creates another fascinating challenge:

**How do all of these agents discover, trust, communicate with, and delegate work to one another?**

That is why I think protocols, identity, governance, and orchestration may eventually become just as important as the models themselves.

# The Future Is Hybrid: Local + Private + Cloud

One of the reasons I continue to find local and private AI so interesting is that increasingly powerful cloud agents don't make local AI less important.

**They may make it more important.**

The future isn't:

**Local AI versus cloud AI.**

The more interesting question becomes:

> *Where should this particular piece of work happen?*

Imagine an agentic environment spanning several computing zones.

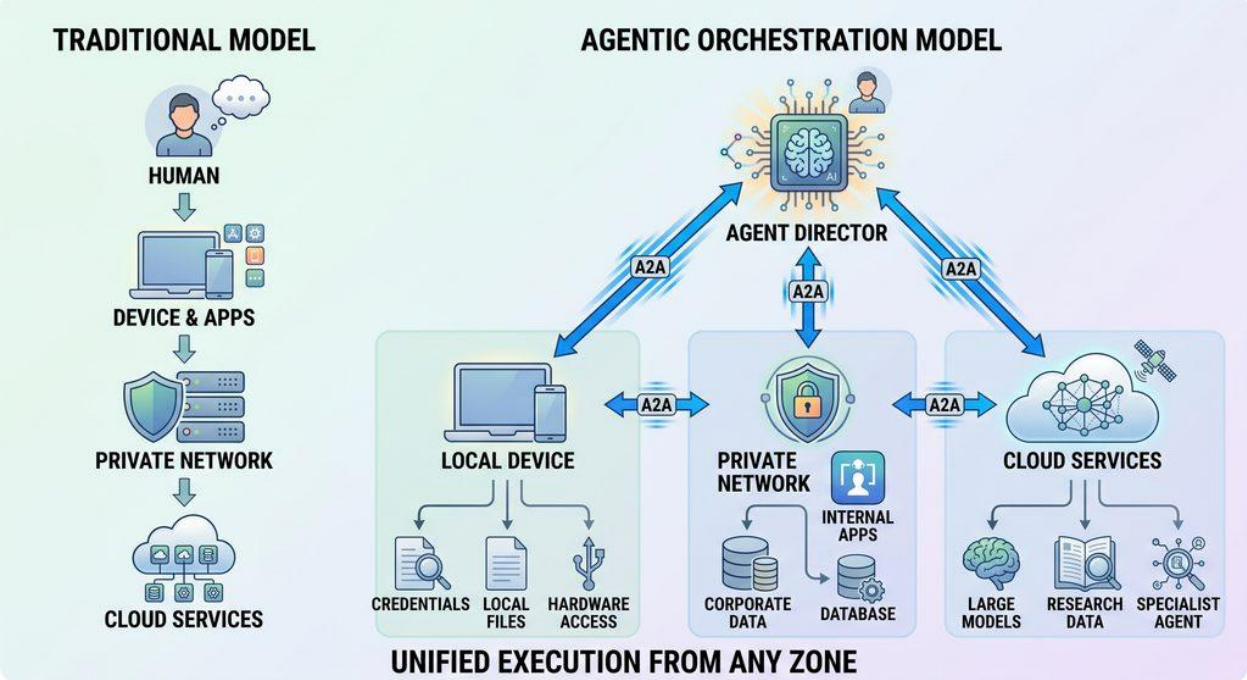

The system could determine where work happens based on:

| Consideration | Likely Environment |
|---|---|
| Highly personal information | Local device |
| Corporate confidential information | Private network |
| Very large model needed | Cloud |
| Low-latency interaction | Local / edge |
| Legacy SaaS application | Cloud computer-use agent |
| Hardware interaction | Local agent |
| Specialized expertise | Remote specialist agent |
| Large-scale research | Cloud |
| Human judgment required | A2H escalation |

This becomes less like choosing an AI application and more like orchestrating computing resources.

# Your Laptop Becomes One Node in Your Agent Network

That leads to another shift I find particularly interesting. Your laptop may stop being the place where your work happens.  Instead, it becomes one node in a larger personal or enterprise agent environment.




Or better yet:


- Some agents may remain active while your laptop is closed.
- Some may never be allowed outside your network.
- Some may use small local models.
- Others may call large frontier models.
- Some may operate applications visually.
- Others may never have a GUI at all.
- Some may communicate primarily with people.
- Others may communicate primarily with other agents.

And somehow, all of this needs to work together.

# From Computer Operator to Agent Director

This may be the part of the transformation that interests me the most.

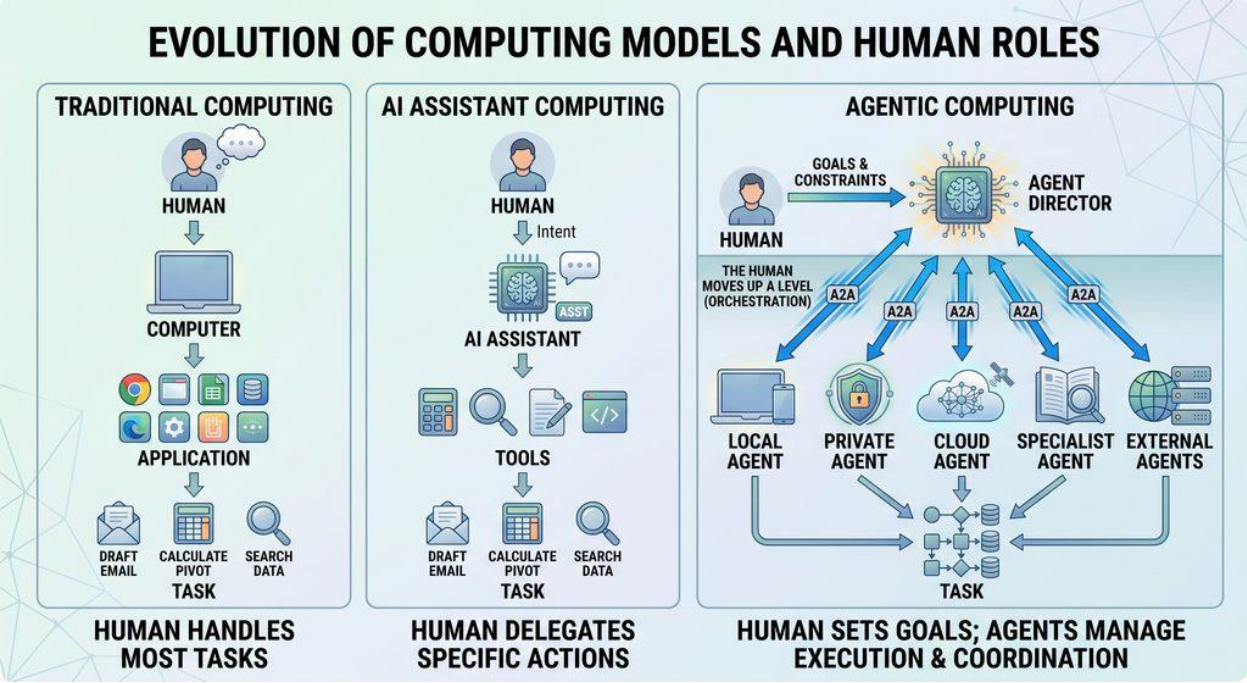
**For decades, the human has effectively been the orchestration layer of the computer.**

We decide:

- Which application opens.
- Where information comes from.
- Where information goes.
- Which document gets attached.
- Which system is updated.
- Which window gets opened.
- What gets copied.
- Who gets notified.
- What happens next.

AI agents increasingly begin moving those responsibilities into software.

That changes our role.




**The human moves up a level.**

Instead of specifying every action, we increasingly define:

- Intent.
- Outcomes.
- Priorities.
- Constraints.
- Permissions.
- Risk tolerance.
- Judgment.

The agents determine how the mechanics of the work happen.

That doesn't make humans less important.

**It changes where humans provide value.**

## This Will Change How Applications Are Built

There is another implication that I think deserves much more attention.

Almost all software today is designed primarily for people.

We create:

- Navigation menus.
- Dashboards.
- Buttons.
- Forms.
- Search boxes.
- Reports.
- Workflow screens.

All because a person needs to understand and operate the application.

But what happens when a meaningful percentage of interaction with software comes from agents?

Applications may increasingly need several different interfaces.

| Interface | Primary Consumer |
|---|---|
| GUI | Human |
| API | Applications and integrations |
| MCP | AI agents accessing tools and context |
| A2A | Other autonomous agents |
| Events / Messaging | Automated systems |
| Human escalation interface | A2H decisions and approvals |

Eventually, a software architect may need to ask two equally important questions:

*How will a human use this application?*

*and*

*How will an agent use this application?*

**That is a profound change in software design.**

## Does the GUI Eventually Become an Agent Interface?

Push that idea a little further.

For decades, the graphical user interface has been the primary abstraction between humans and computers.

We created:

- Menus so people could find capabilities.
- Forms so people could provide structured data.
- Dashboards so people could understand systems.
- Icons so people could recognize actions.

But agents may not need those abstractions.

A mature agent may simply need to understand:

- What capabilities exist?
- What information is available?
- What permissions do I have?
- What outcome am I trying to accomplish?
- What other agents can help?

In some applications, agents may use APIs or structured protocols.

In others, they may operate the GUI.

Eventually, some systems may be designed primarily for agents, with the human interface becoming the secondary experience.

That is when the implications of an agentic operating system become much bigger than another AI feature.

## Identity and Governance Become Critical

Of course, giving autonomous agents computers, credentials, tools, and access to other agents introduces an entirely new category of questions.

- Who owns the agent?
- What identity does it have?
- What systems can it access?
- Who delegated the task?
- What information can it share with another agent?
- Can one agent authorize another?
- Which actions require human approval?
- How long should permissions remain active?
- What happens when an agent is no longer needed?
- What happens when its human owner leaves an organization?
- Can we reconstruct what an agent did?
- Can we stop it?
- Can we revoke access immediately?
- Can we distinguish an approved enterprise agent from an unknown one?

These are no longer theoretical questions.

As agents become more autonomous, governance cannot simply be added later.

We will increasingly need to manage agents much like we manage users, applications, workloads, and services today:

**Identity + Permissions + Policy + Lifecycle + Observability + Accountability**

Perhaps even more carefully.

Because agents don't simply access information.

**They increasingly act on it.**

## I Don’t Think the GUI Is Going Away

None of this means that graphical applications, keyboards, browsers, or traditional interfaces disappear.

Technology transitions rarely happen that way.

The command line didn’t disappear when graphical operating systems arrived.

Desktop computers didn’t disappear when smartphones arrived.

Local computing didn’t disappear when cloud computing arrived.

Each new abstraction changed how much of our interaction occurred at the previous layer.

I think AI agents will do something similar.

We will still use applications.

We’ll still click buttons.

We’ll still open browsers.

We’ll still work directly with computers.

But increasingly:

**we may not be the only ones doing it.**

Our agents will too.

## The Bigger Transformation

This is why I think focusing exclusively on the newest AI model misses part of what is happening.

Better models matter. Reasoning matters. Context windows matter. Inference performance matters.

But around those models, we are assembling something potentially much more transformational:

- Models that reason.
- Agents that act.
- MCP connecting agents to tools and data.
- A2A connecting agents with other agents.
- A2H bringing people into the loop when judgment is required.
- Computer-use systems allowing agents to interact with existing software.
- Local AI providing trusted environments for private data.
- Cloud computers giving agents persistent places to work.
- Operating systems increasingly treating agents as native capabilities rather than simply applications running on top of them.

Put those pieces together and the result is no longer simply an AI assistant.

**It begins to look like an agentic operating layer spanning local devices, private infrastructure, cloud environments, applications, people, and other agents.**

# The Operating System Is Becoming Agentic

For most of computing history, software waited for us.

We opened it. We navigated it. We operated it. We told it what to do.

AI assistants changed that relationship by allowing us to communicate with computers through natural language.

Agents are changing it again by allowing us to delegate work.

Multi-agent systems take another step by allowing the computing environment itself to determine how, where, and by whom that work should be performed.

That means the most important interface of the future may not be:

- a desktop
- a browser
- an application
- a search box
- or even a chat window

It may simply be:

> *What are you trying to accomplish?*

Behind that question could eventually be an entire agentic environment:

- Local agents.
- Private agents.
- Cloud agents.
- Specialized agents.
- Large models.
- Small local models.
- Applications.
- APIs.
- Cloud computers.
- Enterprise systems.
- Other people's agents.
- And humans.

All collaborating around an objective.

That changes the role of the operating system.

It changes the role of applications.

**And perhaps most importantly, it changes our role.**

We move from operating computers to increasingly directing computational resources and digital workers around outcomes.

The transition from computer operator to agent director is still very early.

Grok Bot, Warmwind OS, Omarchy, OpenClaw, Hermes, Agent Zero, ChatGPT Work, MCP, A2A, A2H, and the many projects emerging around them are only early examples.

Many will change. Some will disappear. New ones will emerge.

But that is almost beside the point.

The larger architectural direction is what matters.

**The operating system is becoming agentic.**

And we may be watching the beginning of one of the most significant changes in human-computer interaction since the graphical user interface.

## What an exciting time to be living through this transformation.

We are still in the very early stages, but the implications are enormous. Individuals will increasingly be able to extend their own capabilities with teams of specialized agents. Small teams may be able to accomplish work that once required much larger organizations. Enterprises will need to rethink workflows, roles, applications, security, governance, and even how work itself is organized.

And this transformation will not happen only at the technology layer.

It will happen at every level of the organization—from how an individual manages a day, to how teams collaborate, to how companies design processes, make decisions, serve customers, and create value.

The organizations that benefit most will not necessarily be the ones with the largest models or the most agents. They will be the ones that learn how to combine people, agents, processes, data, and technology in new and thoughtful ways.

That is what makes this moment so fascinating to me.

We are not simply adding another technology to the workplace.

**We are beginning to redefine the relationship between people and computers—and potentially the way work gets done itself.**

**What an exciting time to learn, experiment, adapt, and help shape what comes next.**