The Last Interface

A speculative essay on software in the age of AI agents

Rafael M. Ehlers · July 21, 2026
rafaelmehlers.com.br/essays/the-last-interface

For the past fifty years, software has been designed around the assumption that somewhere, on the other side of the screen, there is a human.

Every design decision follows from this premise. We obsess over navigation bars, buttons, forms, dashboards, onboarding flows, and empty states because we assume that a person will click through them. The interface is not merely a layer on top of the software. It is the product.

But what if that assumption is beginning to fail?

We describe large language models as tools for generating text. Yet their most important consequence may have nothing to do with writing. It may be that they are becoming the primary operators of software itself.

Today, millions of people already ask ChatGPT, Claude, or Codex to write scripts, automate workflows, query databases, analyze spreadsheets, and build small applications. What used to require learning a dozen different products increasingly requires only describing an intention in natural language.

This trend has fueled predictions of a “SaaS apocalypse”: if an AI can build exactly the software I need, why would I continue paying for generic software built for everyone?

I believe this framing misses the deeper shift.

The real question is whether the interface stops being the center of software.

Software has already gone through at least two major eras.

The first was software built for computers. Early operating systems, assembly languages, and command-line interfaces expected humans to adapt themselves to machines.

The second was software built for humans. Personal computers, graphical interfaces, smartphones, and SaaS products all revolved around making computers adapt to people instead.

We may now be entering a third era.

Software built for agents.

Notice what changes.

The human does not disappear. The human simply moves one step away.

Instead of opening ten different applications, navigating ten different interfaces, and manually transferring information between them, the user delegates the task to an intelligent agent.

The agent becomes the immediate consumer of software.

An AI agent has no interest in beautiful dashboards.

It never admires a polished sidebar.

It never gets confused by an onboarding wizard.

It never needs a search box because it already knows what it is looking for.

An agent wants something much simpler.

Capabilities.

Instead of opening a CRM, it wants to create a customer.

Instead of browsing an accounting platform, it wants to retrieve an invoice.

Instead of navigating project management software, it wants to update the status of a task.

To a human, software appears as interfaces.

To an agent, software appears as actions.

For decades, software companies competed by building better user experiences.

How many clicks does it take?

How intuitive is the navigation?

How beautiful is the interface?

These questions remain important for humans.

But if agents become the primary operators of software, a new set of questions emerges.

How discoverable are the available capabilities?

How predictable are their behaviors?

How much context is required?

How recoverable are failures?

How easily can another intelligent system compose these capabilities into larger workflows?

The product is no longer evaluated solely by humans.

It is evaluated by other intelligences.

Seen from this perspective, APIs stop being implementation details.

They become the product.

The graphical interface becomes one client among many.

Perhaps even a secondary one.

This inversion feels strange because it challenges decades of software thinking.

We have long believed that interfaces are what software ultimately is.

Perhaps they were merely translations.

Translations between computational capabilities and human cognition.

If another kind of cognition enters the picture, perhaps the translation changes as well.

This also changes what it means to build a software company.

Today's SaaS products are designed to attract users.

Tomorrow's platforms may be designed to attract agents.

Being the easiest product for an AI to understand, trust, and orchestrate could become as important as being the easiest product for a person to learn.

Documentation, schemas, authentication, permissions, semantic consistency, and machine-readable capabilities become part of the customer experience, not because humans suddenly care more about them, but because the software's first customer increasingly does.

This is why I suspect the phrase "SaaS apocalypse" misunderstands what is happening.

Software is unlikely to disappear.

Businesses will still need payments, accounting, scheduling, databases, identity management, logistics, analytics, and communication.

What may disappear is the assumption that these capabilities must always be consumed through a graphical interface.

The interface survives.

It simply ceases to be the center.

We tell the history of computing as a story of increasingly friendly interfaces.

Perhaps the next chapter is not about making interfaces even better.

Perhaps it is about making them less necessary.

The future may not belong to software with the most beautiful screens.

It may belong to software whose capabilities are so well defined that another intelligence can use them without ever opening a window.