International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

Prefabricated Enterprise AI: Can We Sell Whole Enterprise Systems the Way We Once Sold Windows 98?

S. Muralikrishnan

A memory of buying an operating system off a shelf, and a considered question on whether AI can now sell a whole enterprise system the same way- plug it in, and it runs.

The earlier era sold a box that installed an operating system, and the world built on top of it without a second thought. The question this paper raises is whether the coming era can sell a box that studies a business, then builds, hosts and runs an entire enterprise system on its own, plugged into whichever cloud the organisation chooses- and whether the AI available today is actually strong enough to be that box.

Abstract: ***This paper opens with a simple memory: watching people queue outside a store to buy Windows 98 off a shelf, carry it home on a floppy disk or a CD, and install an operating system that then decided everything built afterward- the language, the tools, the architecture. Java broke that lock-in by promising to run anywhere, on Windows, Unix or Linux, freeing organisations from being trapped inside one vendor's world. The question this paper raises is whether AI can now do the same thing, one level higher-not sell an application, but sell a whole enterprise system, prefabricated and plug-and-play, that a business can buy the way it once bought an operating system, hand a URL and a short form to, and watch it build, host, run and look after itself for years, across whichever cloud it chooses rather than being locked to one. The paper works through this idea with an AI Retail Enterprise package as a concrete example, tests it against a 10% human-intervention bar, and examines honestly what today's AI can and cannot do.***

Keywords: Packaged software, operating-system lock-in, prefabricated enterprise systems, cloud portability, AI-assisted development, human intervention, Frontier AI, autonomous operations

1. The Old Precedent: When the Operating System Dictated Everything

A couple of decades ago, learning to code did not hand an engineer a working system. The engineer had to understand the operating system being built for, because it decided almost everything downstream: the language, the tools, even how the database was shaped. Windows 98 illustrates this best. It was bought off a shelf, installed from a floppy disk or a CD, and everything built afterward- Word, Excel, PowerPoint, and eventually custom business software- ran inside that same Windows world. Visual Studio itself ran on Windows and was built for Windows. VC++, VB, and later VB.NET and ASP.NET, were all tightly bound to the Microsoft stack. On IBM's S/390 mainframes, COBOL was the dominant language, with its own conventions that developers simply followed. Once an organisation picked its operating system, it had effectively picked its language family, its database patterns and its architecture too. Java broke that pattern- a single language able to run on Windows, Unix or Linux- and gave organisations, for the first time, the freedom to change their foundation without rebuilding everything on top of it.

2. The Thought Experiment: A World with No Operating System

A useful thought experiment follows from this. What if there had never been an operating system at all? Every person and every organisation building a computer would have had to invent its own way of managing memory, storage, input and output, before writing a single line of an actual application. There would have been no shared starting point- every computing effort reinventing the same foundation. The operating system spared the world exactly that. It became a layer everyone could simply rely on, so that people could build spreadsheets and business logic instead of first solving how a computer manages its own resources. This is where the idea explored in this paper originates. Today, an organisation that wants an enterprise system still has three broad choices: build it, assemble it, or retrofit it- and all three assume that people, whether employees or suppliers, will spend months, sometimes years, doing that work by hand. The question this paper raises is whether a prefabricated enterprise layer, delivered as an AI package, could do for whole business systems what the operating system once did for computing itself.

3. The New Question: The Organisation as the Customer

A vast ocean of enterprise products already exists- platforms, frameworks, low-code tools, and vertical software for practically every industry. Yet almost none of them can be put to use without a human touch at nearly every step, starting with the most basic question: does this particular product actually solve this organisation's need? The idea explored here inverts that picture. The organisation itself becomes the customer, in the same way a household walks into a store to buy a washing machine, a television, or, in 1998, an operating system. Instead of relying on employees or external suppliers to spend a couple of years assembling, customising and integrating a system, the organisation would shop for the whole enterprise system, ready-made for its industry- an AI Banking Enterprise package, an AI Healthcare Enterprise package, an AI Retail Enterprise package, and equivalents across Communications and Media, Technology, Life

Sciences, Manufacturing and Logistics, Hospitality, and beyond.

Consider an organisation with the idea to establish an online retail store. Under this vision, it would purchase an AI Retail Enterprise package and simply place its content in a cloud space. From that point, everything required to get the system live- architecture, code, data model, integrations, hosting, security, go-live- would be handled by the package itself, without requiring credentials along the way. Where the organisation already runs a legacy system, the same principle applies: the package studies it using only what has been provided. The moment the package requires a human to hand over a credential, the premise breaks down, because the entire model depends on a system that can act without needing to ask.

4. What "Prefabricated" Has to Mean

For this to be more than an appealing phrase, the package would have to accomplish several genuinely difficult things on its own: read and understand the organisation's current landscape, including systems built years ago on technology long since abandoned; work through whatever documents the organisation's own subject-matter experts provide; put pre-fabricated code into action against everything it has learned; test itself thoroughly; check its output against what is already running; self-correct wherever its first attempt does not match expectations; and only then hand the result to humans for UAT and a final sign-off. That last step remains non-negotiable, since it is humans who will use the system and depend on it every day.

Because requesting credentials at every step undermines the model, the practical answer proposed here is a single form, completed once, at the point of purchase — not unlike the questions a customer answers before buying a washing machine: household size, water pressure, available space, intended usage. A new organisation would supply its business idea, its scale and its expected user base, and the package would build the system from that alone. An organisation with an existing legacy system would add one further detail: its public URL. It is worth being candid about the limit of this design, however. A public URL reveals a system's front end, not what sits behind a login screen or inside a private database. Genuine modernisation of a legacy system's real logic and data would still, in practice, require some form of authorised access, even if granted only once. The form-only model works cleanly for a system built from a blank page; whether it extends to a system's private internals with no access granted at all remains an open question this paper does not claim to resolve.

Packaged software then, versus the prefabricated vision now

Dimension	Packaged software era (Windows 98 and after)	Prefabricated enterprise AI (proposed)
What was sold	An operating system, and applications built to run inside it.	A complete enterprise system, by industry vertical: Banking, Healthcare, Retail and similar.
How it was installed	Floppy disk or CD, installed manually by a person, step by step.	Placed in the cloud; the package installs, configures and assembles itself.
What decided the architecture	The operating system and its native language family (VC++, VB, COBOL, and so on).	The AI engine inside the package, reading the organisation's landscape and intake form.
Where it could run	Tied to one operating system, until Java made it portable.	Meant to be portable across clouds — AWS, Azure, GCP, or private — not locked to one.
Testing and acceptance	Tested by the vendor before release; installed and used as-is by the buyer.	Self-tested by the AI, then routed to human UAT and a human sign-off before production.

5. Full-Lifecycle, and Cloud-Agnostic by Design

This vision does not end once the system goes live. The package would need to host it, provision access by role, and then keep it monitored, maintained, secured and running around the clock, year after year, without that operational weight returning to the organisation's own staff. This is a considerably larger undertaking than the build itself; it asks the package to function less like a one-time project and more like a utility that is simply plugged into, in the way a household does not run its own power station merely because it consumes electricity.

The Java precedent is instructive again here. Java's real achievement was not the language itself but its refusal to be owned by a single operating system. A prefabricated AI enterprise package would need the same discipline, one layer up: it cannot be a package that functions only on AWS, or only on Azure, or only on Google Cloud. Binding it tightly to one cloud vendor would simply reconstruct the old lock-in under a different name. The package would need to be cloud-agnostic- fused with whichever environment an organisation already trusts or prefers, drawing on a plentiful choice of cloud environments rather than being tied to a single one. That portability is what would make this proposal a genuine parallel to Java, rather than merely a parallel to Windows 98.

6. The 10% Test, and Where It Gets Hard

The same test used to determine whether an AI-driven lifecycle has genuinely changed anything can be applied here: how much of the journey from purchase to a working, production-ready system takes place without a human doing the work? Under this vision, the human touchpoints narrow to three- completing the form, granting one-time access should private internals need to be reached, and the final UAT sign-off. Should that division hold in practice, the 10% figure becomes a meaningful target rather than a slogan.

Several obstacles stand in the way, however. A URL and a form reveal only the surface of a system, not what sits behind a login screen; genuine modernisation would still require some form of authorised access. Every enterprise's history is different and often poorly documented, and a form completed once may not capture all of it. Regulated industries such as banking and healthcare carry compliance obligations that a self-correcting package would need to satisfy in full, not

approximate. Should an autonomous build, or a system running unattended for years, make a costly error, the question of accountability remains unresolved. Businesses also do not remain static- new regulations, new products, new scale- and a package intended to remain autonomous over the long term would need to keep re-adapting itself well beyond go-live. Current AI models, moreover, are still being proven on long, unattended, enterprise-scale builds and years of autonomous operation, which represents a materially higher bar than well-scoped coding assistance.

7. The Cost Question: Would Only the Billion-Dollar Organisations Afford It?

Should a package of this kind be built, the natural next question concerns cost, and who could afford it. History offers a reasonable precedent: mainframe-era enterprise software was priced for large organisations long before personal computers made software broadly affordable, and enterprise licensing today still tends to follow the same tiered pattern, with the most capable, most customised offerings positioned at the top of the price range. A package capable of absorbing an organisation's entire landscape, correcting itself against it, and then carrying years of hosting, monitoring and security thereafter would require substantial compute, evaluation and support investment- costs that, at least initially, would likely sit closer to what large enterprises can absorb rather than smaller businesses. Over time, as with most technology categories, a tiered market seems plausible: a lighter offering for smaller organisations, and a fuller, more autonomous "Rolls-Royce" version- complete, with nothing left to add piecemeal- for organisations willing to pay for that level of autonomy and assurance. Whether that democratisation occurs quickly remains an open question rather than a certainty.

8. Are Today's Models- Claude, Gemini and Others- Strong Enough?

This is the one question that cannot be answered with confidence, and it warrants an honest acknowledgement of that fact. Today's frontier models are genuinely capable at generating code, proposing architectures, and operating across longer stretches with reduced supervision. What none of them has yet demonstrated, in any broadly proven way, is the capacity to absorb an unfamiliar landscape from nothing more than a URL and a form, construct a fully re-architected production system, and then keep that system monitored and secured autonomously for years, with human involvement confined to the start and the end. This is a considerably larger and messier undertaking than well-scoped coding assistance, and several of the roadblocks outlined above are not purely technical in any case; some are organisational, others legal, and no advance in model capability removes them unilaterally. It is reasonable to believe this could arrive within five years, perhaps sooner, given the pace of change in this field- but that remains a hypothesis to be tested, not a capability that exists today.

9. Testing the Claim Instead of Believing It

Rather than accept this vision as either inevitable or impossible, it is fairer to test it against a few direct questions:

- Landscape understanding: Can the package reconstruct a business from a URL and a form alone, without a human explaining it first?
- Autonomy: What share of the journey is genuinely unattended, beyond the form and the final sign-off?
- Lifecycle ownership: Does the package continue monitoring and securing the system for years, or does that responsibility quietly return to the organisation?
- Portability: Does the package run across clouds, or does it function only within one vendor's environment?
- Evidence: Is the claimed autonomy demonstrated through real deployments and real numbers, rather than a controlled demonstration?

10. What Already Exists, and What Is Still Missing

None of this is being proposed in a vacuum. Legacy-aware tools such as OpenLegacy already map dependencies inside mainframes and generate production-ready APIs in days rather than months- though always through granted credentials, never from a URL alone. Vertical, industry-packaged AI is a real and fast-growing category: enterprise spending on it roughly tripled in 2025 to approximately $3.5 billion, with healthcare leading at around $1.5 billion and legal at approximately $650 million, and Gartner forecasts that 40% of enterprise applications will carry task-specific agents by the end of 2026, up from under 5% a year earlier. Harvey in legal, and Abridge and Hippocratic AI in healthcare, are effectively selling packaged intelligence for a single domain- but as a subscription with an onboarding process, not as a bought, self-installing product. Enterprise vendors are also describing AI-driven development lifecycles in which agents assist with build, test and security inside CI/CD pipelines, deliberately stopping short of full autonomy.

What remains genuinely missing from all of this can be summarised in five points: a retail, shrink-wrap method of purchase; discovery from a URL and a form rather than credentials; build autonomy pushed to the 10% threshold; operation that continues autonomously for years after go-live; and cloud portability, so that the package does not quietly reconstruct the old lock-in under a new name. No current offering appears to combine all five- and in most cases, even a single one of them, on its own, is not yet on offer.

11. Conclusion

The Windows 98 memory, together with the deeper thought experiment of a world without an operating system, matters because both illustrate how completely a prefabricated layer can change what everyone above it is able to build — and how quickly that layer, once established, is taken for granted. Whether AI can now occupy that same role a level higher, allowing an organisation to shop for a whole enterprise system the way a household once shopped for an operating system, is a question genuinely worth asking, given how

rapidly AI capability has advanced. The honest position today, however, is that this remains a forward-looking vision rather than a present reality. The technical, access, regulatory and cost obstacles are real, and a stronger model alone will not resolve all of them. Whether Claude, Gemini, or whatever succeeds them, ever reach the point where an organisation can submit a form and a URL and receive in return a fully built, hosted, monitored and secured production system running on whichever cloud it prefers, is something that should be judged by evidence over the coming years- time to production, quality, adoption, durability, and affordability- rather than assumed in advance.

The store that once sold an operating system on a disk changed what engineers could build on top of it. Whether a future store can sell a whole enterprise system the same way- plugged into any cloud, not locked to one- will be decided by evidence, not by the label "AI" alone.