AI Cambrian Explosion **in Software**

Ali Pichvai
Co-Founder of Quod Financial

March 2026

As AI progress accelerates, there is a tectonic shift manifesting not only in the public markets but also in the debate over what it means for society and the knowledge economy. The reaction in the public markets is essentially to sell off all sectors which are directly exposed, such as SaaS companies, or indirectly exposed, such as office REITs, driven by the fear of being too late to exit these investments. Uncertainty about the impact of AI on various sectors is, in turn, hitting private equity and the credit markets, quickly closing investment and refinancing.

As enough has been said and written about the overall trajectory of AI, I will narrowly focus on its impact on the business-to-business segment of the software industry, simply called software.

I base my essay on the following assumptions and constraints:

- **Artificial General Intelligence (AGI) is not around the corner:** Whether viewed as an imagination-grabbing marketing tool or a Sci-Fi-obsessed AI community mantra, AGI is not close to reality and will not be built on current foundational language models. Instead, what is currently available is built upon a vast written knowledge base, which represents only an infinitesimal fraction of human—or even animal—data and signal processing capabilities. Ironically, this means that the “human expertise” we once praised, gleaned from years of study, has become the primary target of AI.

- **Vibe coding is not fit for complex software:** Vibe coding is simplistic and mostly good for standalone development. It cannot and will not replace the engineering required for complex software that is highly integrated with different workflows and logic.

- **Predictions about the software industry are based on a short-term view (3 to 5 years):** We are, in fact, in the midst of an exponential progression, which means predictions can diverge massively if a small change happens in the overall conditions. This is the nature of non-linear functions, where small changes in initial states lead to major changes in future outcomes. Furthermore, second-, third-, or higher-order impacts, often in the form of feedback loops, are hard to foresee. For example, the software industry is part of the wider economy, and large-scale AI-induced unemployment can itself depress consumer demand, hitting the economy and, consequently, business activity and in turn the software industry.

Simple taxonomy

To paraphrase Marc Andreessen, software has already eaten the world—but it has done so in evolving forms. The traditional division of B2B software into “Horizontal” (industry-agnostic) and “Vertical” (specialized sectors like Law or Fintech) requires a new framework. I instead categorize software by its density of Human knowledge, Structural complexity, and Heterogeneous interaction. Under these axes, a new division emerges:

1. **The Aggregators (Data-hubs):** These systems collect and normalize internal data (e.g., a CRM) or external data (e.g., a payroll system). They provide standardized workflows and, while often “low” on embedded human knowledge, vary from low complexity to high for large organizations, such as a Fortune 500 ERP system.

2. **The Logic Providers (Decision-engines):** These entities provide the “brain”—such as pricing engines or risk management systems. They represent deep, evolving human expertise, requiring high complexity but mid-level integration.

3. **The Integrators (Glue):** In an API-driven world, these providers manage machine-to-machine communication. But not all integration is digital, and part remains human. While the documentation may be poor, the core human knowledge required is lower, focusing on technical protocols of low-to-mid complexity.

Modern providers can embody all three roles, simultaneously being shaped by and shaping corporate structures. Ultimately, how human knowledge is acquired and maintained will soon be defined by AI.

AI (Agent) versus Software

An AI Agent is the orchestration of AI components (such as LLMs) and human interaction, utilizing an architecture designed to execute a series of steps that probabilistically determine the most correct set of actions. For example, a CRM Agent doesn’t just “talk” about customers; it actively uses tools to update a lead’s status within the system.

The software sector is certainly the first business to be hit in an almost perfect AI storm; software engineering will rapidly change as coding is a simple natural language, with a low level of “fuzziness”, and comes with a massive amount of available structured and unstructured data. The question is then whether software is going to be necessary or will be replaced by AI and AI Agents, which will be self-coding and self-organizing. Or is AI a productivity gain tool, automating part of knowledge-based activity, from repetitive work to expertise.

AI comes with a certain number of underlying foundational constraints:

- **Deterministic versus Probabilistic:** The shift from deterministic, rules-based software to the probabilistic nature of AI represents a fundamental change in how future systems will be engineered. For instance, would you want to risk leaving a critical decision to a probability tree? If you are executing a $100M market order, you require the certainty of a deterministic system, rather than the "probabilistic guess" of an AI prompt.

- **Context window and memory:** The context window—the maximum amount of information an LLM can hold in a given session—is one of the biggest limitations of current technology. Because a model cannot "remember" more than its maximum token limit, it often suffers from forgetfulness and hallucinations as the session's tokens are consumed. Paraphrasing Dario Amodei in his essay, The Adolescence of Technology, we are facing a nation of Nobel Prize winners who are afflicted with severe Alzheimer's disease. A short-term solution involves the set of techniques known as Context Engineering. In the longer term, there must be a substantial increase in available memory. AI is currently in a situation similar to the early days of the internet with its limited bandwidth (which required incredible investments in fiber optics to fix), or the early car industry navigating poor roads that were only adequate for horse-drawn carriages. Ultimately, this means that complexity is currently much harder to handle and can lead to unexpected, adverse results.

- **Explicit versus implicit knowledge:** Companies operate using a certain level of explicit knowledge—the information they actively write down and codify. However, a large percentage of knowledge is never recorded; instead, it relies on implicit institutional memory and past know-how. It is this unwritten understanding that allows companies to operate and succeed in an ever-changing world. Because this "data" set is missing from formal records, it will always require flexibility and a human presence. In this context, AI may know the universe, but not the atom.

An additional aspect to consider is the current performance (load/latency) requirements for certain critical systems, which cannot be properly handled by pure AI or AI agents. A flight management system managing hundreds or thousands of planes, and hundreds of thousands of events, operates under very stringent performance demands. In these cases, AI agents will not replace purpose-built software in the medium term.

Taking these considerations into account, current AI agents will handle a portion of the software workload—such as integration systems managing a multitude of API connections—that is not mission-critical. They will also be capable of managing simpler aggregator workflows. However, the best outcomes are achieved when purpose-built software is coupled with AI to provide:

- **Automation**: AI will provide a layer of automation around complex software, reducing setup and maintenance friction while enhancing intricate workflows. This will primarily be achieved using open-architecture software with well-documented APIs and transparent underlying processes. Ultimately, this will trigger a much-needed productivity gain for knowledge-based activities, which have been stagnant for some time.

- **Redesigning processes:** Historically, software-imposed processes have often forced clients into rigid and unnatural workflows. Besides the positive impact AI is having on the software industry—enabling vendors to accommodate a wider variety of client-specific use cases—AI itself can serve as the backbone for new, flexible processes that leverage traditional software for critical steps.

- **Reporting and Transparency:** Traditional software often operates as a black box, with strict, static limits on what information can be extracted. AI greatly expands reporting capabilities, generally providing much greater transparency into the inner workings and decision-making processes of the software.

A software company is more than coding

AI will introduce new automated coding practices to software engineering, which will increase productivity, lower time to market, and improve maintainability, ultimately freeing up developer time to focus on more valuable activities. Even if a specific target cannot be defined, it is reasonable to expect that automation levels will increase to cover around 80% of development activity in the near future.

- **AI Coding:** A New Paradigm: The most common use of AI in coding centers around assisting developers with tasks like writing code, creating unit tests, or drafting documentation. This is a human-centric view of what AI can do. The real shift is to flip the logic and place AI at the center of the development process. However, this must be done while acknowledging the constraints of current AI technology: the inability to “tokenize” very large and complex legacy codebases, small context windows, the potential for incorrect code (even with beautiful syntax), and the difficulty of managing implicit knowledge. In this new approach, engineering is organized as a process designed to plan, code, review (both automatically and by humans), and test the generated code. Testing, in fact, becomes the most critical part of overall software development, encompassing as many dimensions as possible. This goes beyond mere unit testing; it must be sequenced to include integration and performance testing before any new code is committed. This is the price to pay to avoid code that looks well-written but fails to grasp the complexity or dependencies of legacy systems. It requires a deeper understanding of what the code actually does, what distinguishes good code from bad code, and how to maintain corporate know-how regarding core functions.

- **Reskilling Your Workforce:** A massive jump in productivity that frees up coding and testing time has two immediate effects on the workforce. First is the overall underutilization of certain coders and testers, which presents both economic and motivational challenges. This will inevitably lead to staff reductions to adjust to the new reality. The second issue is that the current engineering structure was built for “manual” coding, relying on a skill set primarily based on technical knowledge and execution. Under this new paradigm, there is less need for technical brilliance—which relies on deep, specialized knowledge—and a greater need for architectural thinking. You need professionals with broad knowledge who understand how the entire system should work and how business functions should be designed.

This means your current engineering workforce must reskill (and be willing to do so). With increased productivity, your next bottleneck will be finding the right people to manage automated coding: **AI engineers** who understand your product, know how to build new features, and recognize how good code functions. Remarkably, you could find yourself with too many “old-style” coders and still be incapable of increasing output because you have too few “new-style” AI engineers.

- **Dreams Come True (Or Almost):** The positive case for AI-driven productivity is that almost every software house has a backlog of features and reengineering tasks—such as rewriting poor-quality or aging software—that were never properly prioritized. This is a time when increased productivity can finally allow you to clean up shop. However, this is largely an engineer’s dream and is, at best, tactical. As I will discuss later regarding the economic impact on the software business, the broader reality is that launching new products has become easier, faster, and more economically viable. This lowers the barrier to entry—which is a good thing for some, but a threat to others.

But a software product business is not solely comprised of coders, testers, and DevOps. In fact, in my own company, these technical roles make up less than 50% of the workforce. The other half consists of people who work directly with clients to onboard them onto our platform, help them evolve the software to meet their changing needs, and support them on a daily basis—24/6 across five continents (with the seventh day reserved for maintenance and upgrades). It also involves interacting and keeping pace with data providers, marketplaces, and various other vendors.

Finally, it requires managing multiple software versions, as core product features are utilized differently by various clients. This includes handling specific customizations that must be frequently upgraded without causing disruptions. It is an issue of immense complexity—one that is already managed by automated tools as much as possible, but which ultimately relies on experience-based human expertise.

Software is not just a tool; it is a mechanism that shapes organizations. We have to navigate clients who have their own distinct corporate cultures, complete with the politics, inefficiencies, and inertia that directly affect the success of a partnership. This means we must understand the corporate sociology of each client in order to succeed (and to avoid the abysmal track record of failed IT projects). So, unless we are all heading toward a future of one-person companies run entirely by AI, sitting on a beach waiting for a universal basic income, the software business will continue to be heavily shaped by these human interactions, constraints and realities.

The economic case in the software Cretaceous period

What is certain is that AI is going to disrupt the economics of the knowledge economy. This includes not only software but also professional services — the lawyers, consultants and coaches — and the sectors exposed to them—from office real estate to Australian-blend flat white & oat-milk Uji matcha coffee shops.

AI and AI agents will directly hit revenue streams, replacing portions of current “simplistic” software businesses, or disenfranchising them as agents become the gatekeepers. But even the more complex, high-performance, and human-interaction-dependent software businesses must face their own reckoning. As AI engineering becomes the de facto reality, it will erode monopolistic (or quasi-monopolistic) positions, shifting bargaining power back to corporate clients. Furthermore, in traditionally “sticky” software domains where clients once never dared to envision migrating away from a platform, they will now be empowered to make that switch. Clients will inevitably demand lower pricing as the cost of building and running software decreases alongside increased automation and productivity gains.

Private Equity has heavily invested in and consolidated the software industry. Stable revenue streams and high EBITDA enabled leveraged buyouts, while exits to public markets or other PE firms created a massive incentive to invest. However, as revenues and EBITDA margins deteriorate, the private credit and bond markets are starting to price in higher risk, potentially morphing into a debt trap. This triggers immediate cost-cutting, leaving little room for transformation or investment in the new world—including the vital reskilling of the staff. Ultimately, it precipitates large-scale layoffs just to comply with debt covenants, further amplifying the crisis.

The next question, then, is: can any company thrive in an ever-decreasing revenue cycle? In many ways, growth is a measure of the future viability and health of any entity. It is almost a human and social expression of the second law of thermodynamics: a company that does not grow is ultimately defined by decay. Growth is not solely an expression of revenue. Rather, revenue is the “energy” that sustains companies—fueling the influx of new ideas, new people, new clients, new products, and new impetus.

> *“I frequently hear the argument that ‘we will no longer need juniors,’ and should only hire senior developers for our engineering teams. This is another trap: an aging organization deskills itself into oblivion. Despite the rise of automation, you must continuously bring a flow of next-generation talent into the organization to survive.”*

Because all of these are human activities, if AI inevitably shrinks companies, adopting a defensive posture simply means accepting a slow—or perhaps quick—death. The alternative is to accept lower price points while offering significantly more products and services to your clients, to keep growing.

This is where AI comes back into play, as the cost, time, and difficulty of bringing new products to market are drastically reduced. This is the bet that software companies must make. However, it also guarantees a hyper-competitive environment characterized by brutal price wars, where companies on the verge of bankruptcy will desperately take on untenable client contracts just to survive.

Corporations must therefore change their evaluation criteria. Where they once considered sheer size and a "strong" balance sheet as proof of vendor viability, they must now view size with suspicion and treat high debt as a ticking time bomb. We are reaching the end of the software industry's Cretaceous period, and only those who adapt quickly to this new environment will survive.

"It's economics, stupid!"

If software is acutely suffering, the AI sector is also facing fierce competition—from tech giants fighting at a level perhaps unseen since the 1830s US railroad mania, to the geopolitical battle between the US and China.

The battle lines are drawn around the following areas:

- **AI Operators and the Current Phase of Investment:** After the opening salvo by OpenAI in November 2022, we now see a multitude of commercial players racing to become the leader. This intensely competitive race is incredibly tight, which has raised the stakes for infrastructure requirements—specifically the GPUs, data centers, and energy needed for training and inference. The hyperscalers have calculated that they cannot afford to settle for number two, as falling behind could mean the immediate demise of their cloud businesses. Meanwhile, the cost of operating these models is increasing, rather than decreasing through technology commoditization and the natural economies of scale that usually accompany higher adoption. This stems from Nvidia's quasi-monopolistic hold on the GPU domain, but also because securing data centers and energy has created intense inflationary pressure. In this development phase, users are barely covering the CapEx and OpEx of the AI operators—neither the cost of creating and training the models, nor the cost of running them (inference). The true promise of Moore's Law is not just rapid processing power, but an exponential decrease in cost. This current dynamic can persist only as long as investors are willing to bankroll startups, and bond and stock markets continue to finance the hyperscalers. Even if we have a few more years, the timeframe is not infinite; the capital required is vastly higher than during the asset-light internet boom.

This signifies that either costs must drop drastically, or the current economics of AI and AI agents will change substantially, resulting in steep price increases. This will dictate how AI engineering and the software business itself evolve. It also means not everyone will survive the next three to five years. Some AI operators will disappear or be acquired by hyperscalers (or better-financed entities), and some current hyperscalers will inevitably throw in the towel.

- **Open vs. Gated (Commercial) Models:** While US players aim to secure the prime position in the commercial marketplace, Chinese companies have decided to bet on open-source (or open-weight) models. Furthermore, under the US stringent chip technology restrictions of the past five years, Chinese firms have been forced to adapt their technology to cheaper, lower-end GPUs, which in turn has created a lower-cost option. This means their open-source/open-weight cost benefits are combined with better operating efficiency, yielding models of quasi-equivalent quality. Yet, these economically efficient Chinese models are a difficult (if not forbidden) proposition for certain clients and jurisdictions, as the US and China remain locked in a trade Cold War.

- **Confidentiality and Know-How Leakage:** The privacy of data shared with an AI provider is a major concern for most corporate clients. Insidiously, one of the hidden risks is know-how leakage. Even if most providers claim that user data is not shared, recent papers suggest that "seen" data inevitably influences the model's underlying "understanding." This means that sharing the intimate mechanics of your business creates a fundamental risk of exposing your raison d'être.

On one hand, there is an imperative for software companies to rapidly embark on a transformation toward AI engineering. On the other hand, they face immense uncertainty regarding the choice of AI models (predicting which ones will survive), the future economics of AI engineering once investor-subsidized pricing gives way to reality, and how to protect proprietary data and know-how.

The only path forward is to remain highly adaptable, maintain a high level of experimentation, and avoid being trapped by a future high-cost, highly aggressive provider.

No apocalypse, but Cambrian explosion

I am a child of the 1979 Iranian Revolution. I witnessed how an elite was forced into exile across the Western world, losing their social standing and economic livelihoods in the process. The sudden rupture crushed those who could not adapt, and very few above a certain age managed to do so. But for us, those who were children or teenagers, this was the new reality, and we rebuilt our lives. What is happening to the knowledge economy today is very familiar. Most people are early in the five stages of grief (the Kübler-Ross model), with some in Denial and others in Anger. But for me, Acceptance has come naturally.

Capitalism's wealth creation is built on a unique foundation: the belief in a better future. This belief generates the trust—or credit—required for economic actors to collaborate, including providing the capital needed to build that brighter tomorrow. The opposite is also true: if we believe the future is bleak, credit dries up, triggering a rapid contraction in economic activity and the destruction of personal wealth.

The first Industrial Revolution made hand weavers redundant, but it also allowed a peasant, who could hardly afford more than a handful of woolen garments in the 18th century, to own a much larger wardrobe, with a multitude of fabrics being shipped across continents. Knowledge workers and the companies that employ them, such as software firms, must now multiply not only their productivity but also their creativity and output. This is the AI Cambrian explosion—a period that will make things possible that we never even imagined. And it will ultimately enrich individuals and society in the process.