THE DISRUPTOR

Agile Is Dead!

Arun Varadarajan
Chief Commercial Officer
Ascendion

Software engineering must evolve, again.

Because we have ventured into a new era of software development where Agentic AI systems don't just assist engineers; they are the engineers. This is the age of Agentic-Driven Engineering.

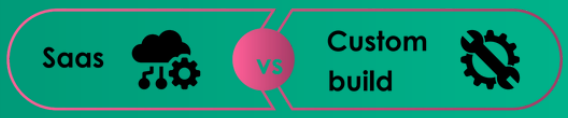
But before we dive into the nitty gritty of this revolutionary stage of engineering evolution, let's take a step back to understand how this will impact how we build software. The waterfall model marked one of the earliest approaches to software development, but it was sequential, inflexible, required heavy documentation, and rigid to change. These waterfall-based projects took so long that by the time the software was delivered, the requirements used were no longer valid.

Agile was developed in response to these limitations. It brought in fast iterations and collaborative methods and was feedback-driven, addressing the condition of changing requirements. It served well, barring the fact that it was highly process-reliant, human-intensive, and ritual-driven and still did not impact the speed to market sufficiently.

Both these models were born in an era where engineering was predominantly human-driven with some automation, making things slow and error-prone, requiring multiple checkpoints and reviews.

The advent of Agentic AI now flips the model. It automates not just the process of coding but decision-making, prioritization, and execution based on data and context with minimal human intervention. It can drive a higher degree of predictability and move the software life cycle into a factory model where software is building software, and the human engineers are dealing with exceptions and things the AI agents are not able to handle. They become the supervisor of these Agentic AI systems, ensuring that the factory produces a good quality product.

Soon, SaaS and packaged goods offerings in the market will no longer be attractive, in my opinion. If I can build custom software cheaper and more tailored to my needs and the IP is mine, why would I need off-the-shelf commercial software, which anyways requires a high degree of customization to make it work for me? This change is imminent.



Let's take a look at how Agentic AI systems can develop applications and outpace agile systems at every stage.

Building an App – Agile vs Agentic

Suppose a solar energy company wants to build an internal app with a clear objective of streamlining sales operations and empowering reps with real-time access to crucial information anytime, anywhere.

With agile methodology, this app would take six to nine months to build. Multiple sprints are just used to harden the requirements and create the backlog. Product Managers would first gather detailed requirements to create epics. Epics would then need to be put into a product roadmap and groomed into actionable stories and tasks. Epics will have to be converted to user stories. Then the sprints get planned across all the app layers - presentation layer, the business logic layer, the data access layer and the infra layer as required.

In a given sprint, UX designers create wireframes that meet user needs. UI developers are converting the design into responsive code. Full stack developers are working on the business logic following standards, which would go through multiple rounds of QA testing. At the end of each sprint, the stakeholders would review the progress and provide feedback, and the cycle of revisions would begin in the ensuing sprint.

As we have all experienced, this process is slow and highly dependent on human coordination plus multiple feedback loops, which leads to the backlog getting moved from one sprint to another, extending the timelines and defeating the purpose of AGILE. It's not AGILE that is the problem, but it's the implementation by human engineers that introduces unnecessary overheads, leading to delays and disappointments.

Now, let's consider how the Agentic AI engineering model will work. We are calling the model Lean Software Engineering, and here's the high-level development process that we are institutionalizing within Ascendion. It uses iterations that are nothing but mega sprints that mimic a conference room pilot in the ERP implementation scenario.

Establishing the foundation: Tailor and customize the AI Agentic system for the client's needs. This process is started by ensuring that the AI Agentic system learns the overall application need, appreciates the tech stack from the presentation layer to the infra layers, absorbs the enterprise standards, learns the overall architecture and landscape, imbibes the design system, obtains seed data from the client for the application, and understands the non-functional requirements. During this phase, Agents and workflows are designed and ready to go, gaining all the necessary knowledge they need to perform. This phase should take 6-8 weeks, depending on complexity.

A series of iterations: The method requires a series of iterations. Each iteration comprises Agent Execution, Agent Validation & Stakeholder Joint Workshop (Human in the Loop). Post the foundation step within each iteration, AI Agents will analyze these inputs, resolve ambiguities, and start generating data models, front-end & back-end architecture, design layouts & UI flows, APIs & integration logic, test cases, and deployment pipelines, all in a coherent manner.

Iteration zero: Run the Agentic workflows across all the application layers with the information obtained in the foundation phase. The engineers in the loop will give the client a Demo Iteration zero. Obtain feedback on the application, document all the changes, and identify agents/ workflows that need tuning and customization. This run should take 1-2 days.

Iteration one: Refine the agents and redesign the workflows to incorporate the feedback. Run the revised Agentic workflows. Engineers are in the loop with demo Iteration 1 to the client for feedback. Once the feedback is obtained, the cycle to fine-tune the Agentic AI system is repeated just as Iteration zero above, and the run effectively moves to the next iteration.

Subsequent iterations may be used to fine-tune the application output. We recommend not more than two iterations post iteration zero to arrive at an MVP.

AI agents can generate a functioning prototype with workflows, pricing logic, and CRM integration in less than 2 weeks after the foundation phase. Stakeholders can keep interacting with the app directly, providing contextual feedback within the interface, and the loop can be closed in the next iteration, unlike Agile, where you are not sure which sprint will address your gaps. Think of each lean software iteration as a mega sprint that fundamentally builds the entire application again and again until we are satisfied. It is similar to running a few trial production runs in manufacturing before going for the full run.

In the Agentic AI model, every aspect of the output is versioned, explainable, and auditable, and it is ready for review, modification, or approval.

Agentic Trumps Agile

There are, hence, clear advantages of Agentic AI systems that trump agile methods:

- Agile works typically in 2-week sprints of a limited set of items to attack. Agentic AI iterations attack the entire application need, so don't wait for planned meetings.
- Agile demands alignment of PMs, designers, developers, QA, etc..It needs them to sync constantly. Agentic systems break this dependency. Each AI agent understands its domain, reads the shared context, and gets to work. It allows for the actors in the lifecycle to examine the deliverables as they get generated and regenerate them if they are not to their standards.
- Agile depends on demos and retrospectives to explain what was done and why, as the client may not see the entire system in play. The Agentic AI method comes with the entire system built so that the client can actually experience the application, allowing for more pertinent feedback. Also, every decision, every change, and every output is tracked, audited, and explained. The level of traceability is extremely high in the Agentic Development method.
- Agile consumes significant human effort. Planning meetings, backlog grooming, story-pointing - it all adds up. Agentic systems need just a well-framed requirement, a design system, enterprise standards, and access to organizational rules. They do the rest and provide the human engineer with the ability to make adjustments and fine-tune things as we progress through the life cycle.

Conclusion

The future will no longer be a room full of whiteboards or calendars filled with standups. Engineering will now be fluid, transparent, and continuous, where you can experience the entire product functionality directly after every iteration. Iterations will be shorter and more meaningful, with sharper alignment to user needs and minimal rework, as standards are followed from the start by AI Agents.

In this new model, clients can focus on business value, outcomes, and the big picture without getting caught up in the minutiae of technical details, which are now handled by AI Agents. The compelling aspect of Agentic AI is that it understands intent, applies constraints, and delivers an experience that works.

So, in the not-too-distant future, Agile will die, and so will SaaS.

If we can build a custom app that is significantly cheaper and faster, with only the features relevant to you, why would you pay license or subscription fees to a SaaS vendor? Especially when you don't use all the features and still spend time and money customizing them?

Moreover, the IP is yours when you build your own!

About the Author

Arun Varadarajan is Chief Commercial Officer at Ascendion, where he leads go-to-market strategy and helps clients build AI-powered digital products that drive real business value.

With a career spanning leadership roles at Cognizant, Oracle, and Capgemini, Arun has delivered over $1B in transformation outcomes across data, AI, cloud, and enterprise platforms. He's built startups, advised global enterprises, and shaped high-impact teams that scale innovation.

Relentlessly curious and future-focused, Arun is driven by a simple goal: help businesses thrive by engineering what's next. He lives in the Dallas area with his family.