A FUTURE VISION

WHY AI-DRIVEN DEVELOPMENT MAKES AGILE SOFTWARE DEVELOPMENT OBSOLETE

Dr. Brian Scott Glassman

Introduction

AI-driven software development is eliminating the relevance of Agile entirely. For twenty-five years, Agile has been the premier framework for managing software development, but CTOs and VPs of Engineering now recognize that its core principles, designed for human-paced development, are being broken by AI models that generate hundreds of lines of code per minute. We are in a transitional period, waiting for a new framework that addresses AI-driven realities. What once made Agile revolutionaryspeed, collaboration, and iterative feedbackhas become obsolete.

Our teams have extensive experience building and creating processes for AI-driven software development tools, and together we have made the following findings. This article explains why Agile's major principles are no longer valid and how clinging to them now hampers progress. We review six key points: delivery speed, daily collaboration, sustainable pace, simplicity, self-organized teams, and customer needs over lines of code. Your team should understand why Agile is no longer appropriate and begin organizing around a new paradigm that embraces AI's capabilities rather than outdated human-centric processes.

1. Delivery Speed: When Agile's Core Strength Becomes Its Fatal Flaw

Agile emerged as a response to Waterfall's glacial pace, with its principle of delivering working software in weeks rather than months. This was revolutionary in 2001. But AI has shattered these timelines completely. Development cycles that once took months now compress to weeks, and weeks to days. AI coding models generate hundreds of lines of functional code per minute, with teams of AI mesh agents soon capable of updating entire codebases simultaneously. Entire projects now complete before the next scheduled sprint review.

When your development process is built around two-week sprints, but AI delivers a complete feature in two days, you're forcing artificial delays into your workflow. The speed of software development is no longer the bottleneck yet Agile processes continue to organize work as if it were. Future systems for organizing work will be driven by AI themselves, not human Scrum Masters. The AI will determine optimal task sequencing and delivery schedules based on actual technical constraints rather than arbitrary meeting cadences.

2. Daily Collaboration: When Human Coordination Cannot Keep Pace

Agile mandated that business people and developers work together daily, assuming human coordination was essential for success. But this was designed for human-paced cycles where daily standups could keep everyone aligned. In AI-driven development, the speed is too much for traditional collaboration rituals. When AI mesh agents coordinate simultaneously across codebases, updating multiple sections at once, code collisions become inevitable and happen far faster than any daily standup could address. AI itself will have to facilitate collaboration between developers, managing merge conflicts, coordinating parallel workstreams, and resolving integration issues in real-time. We need AI systems that mediate between developers, prevent collisions before they occur, and orchestrate work at machine speed while humans provide strategic direction rather than tactical coordination.

3. Sustainable Pace: When Human Endurance Is No Longer the Constraint

Traditional Agile emphasized sustainable pace to prevent developer burnout. Teams were expected to move at a steady, manageable pace measured by story points and burn down charts, treating software development like an assembly line where human capacity determined output. But with AI, velocity becomes completely uncoupled from human effort. A development team can now build more software in two weeks with AI than the same team of humans could produce in six months. The constraint is no longer pace or human stamina. Instead, limiting factors shift to code quality, customer fit, application performance, AWS operational costs, or AI inference bills. Story points and burn down charts become irrelevant metrics when AI does the bulk of coding labor. The new challenge is managing the constraints that emerge when coding speed is effectively infinite.

4. Simplicity and Self-Organized Teams: When Less Is No Longer More and Size No Longer Matters

Agile championed simplicity, maximizing the amount of work not done, because every extra line of code meant more time, risk, and human labor. Similarly, Agile believed self-organized teams closest to the work would design better systems. But AI obliterates both principles simultaneously. With AI writing code, the marginal cost of generating extra modules or multiple user interfaces is minimal. Teams can mock up several UI options and cross-compare them with AI-simulated customer reviews, openly violating the simplicity principle but producing better results.

The real cost is no longer tied to lines of code but to optimizing the codebase to work effectively with AI and applying best practices to prevent catastrophic errors like when an AI deletes key functions while working outside context window limits. Meanwhile, team size is shrinking dramatically. A few experienced developers with high-performance AI models can accomplish what once required entire departments. Simplicity as a constraint and self-organized teams as a structure both fade when AI changes the fundamental economics of code production.

5. Customer Needs Over Lines of Code: When Customers Become the Bottleneck

Agile prioritized satisfying customers through early and continuous delivery, but assumed gathering customer feedback was faster than building software. AI has inverted this relationship. Development now moves so fast that entire projects complete before the next scheduled customer feedback session, making live customer input the new bottleneck. AI-powered systems can simulate hundreds of rich customer interactions from comprehensive personas, accelerating feedback cycles and generating actionable insights instantly. More critically, AI can collaborate with customers, product managers, and stakeholders to uncover the true issues driving change requests, because customers often provide limited or convoluted feedback requiring investigation to identify core intent. AI can validate whether changes are necessary, brainstorm holistic solutions, and calculate broader impact across the entire codebase. The fundamental shift: lines of code are no longer expensive or time-consuming, but understanding genuine customer needs remains difficult and slow. When coding capacity becomes effectively infinite, the entire Agile framework collapses. What matters now is using AI to deeply understand customer intent and deliver transformative value rather than surface-level feature requests.

Conclusion

Agile served its purpose for a quarter century, but AI-driven software development has fundamentally broken every assumption it was built upon. Speed, collaboration, pace, simplicity, team structure, and customer feedback cycles have all been transformed beyond recognition. The industry now demands a new framework and philosophy for managing software development, one designed from the ground up for AI capabilities. Our teams and thought leaders are actively working to create this next framework for AI-driven software development that will fuel the next decade of progress. Forward-thinking organizations must begin building and adopting these new practices now or risk becoming as obsolete as Waterfall.