Scrum.org™ The Home of Scrum | Whitepaper

Empiricism at Machine Speed

Part 5 - The Developer's Nexus: *What Happens to Scrum When Every Team Member Governs Agents*

Matthew Hodgson, CEO, Zen Ex Machina & PST
Dave West, CEO, Scrum.org

part 5 of 5

September 2026

© 2026 Scrum.org | All Rights Reserved.

Executive Summary

By 2028, an average global Fortune 500 enterprise will have more than 150,000 AI agents in use, and only 13% of organizations think they have the right AI agent governance in place (Gartner, 2026). The difficulty is structural rather than technical: where does the human sit when AI produces more work than any one person can review?

Papers 1–4 of this series argue that Scrum's empirical pillars, transparency, inspection, and adaptation, address the three governance gaps that AI agent systems create: who is accountable for what gets built, who decides whether it is valuable, and what the system's struggles reveal about the organization (Hodgson & West, 2026). That argument drew on twelve months of operational evidence from the Relentless Raptors, a single hybrid human-agentic Scrum Team in which one human Product Owner governs AI agents working as the Developers. The question the Raptors leave open is what happens when a Scrum Team has more than one human Developer, and each of them governs agents.

The Scrum Guide allows no sub-teams within a Scrum Team (Schwaber & Sutherland, 2020), so a Developer's agents cannot become a team inside the team. Nexus already solves the problem of several teams building one product, and it does so by adding a small number of elements to Scrum without changing it (Schwaber, 2021). Applied here, each human Developer's agents form a team of their own, working from the one Product Backlog under the one Product Owner, and the human members of the Scrum Team take on the accountability of the Nexus Integration Team: producing one Integrated Increment that meets the Definition of Done. That structure is the Agentic Nexus. Because the pattern is borrowed from a framework built for human teams, whether it holds for AI agents is a question practice will settle, and the paper names its boundaries accordingly.

The Scale the Scrum Community Is Heading Toward

Gartner's forecast describes agent sprawl: different teams building agents without centralized governance, consistent oversight, or any inspection rhythm (Gartner, 2026). In Scrum terms, most of those agents sit outside any Product Backlog, answer to no Product Owner, and meet no Definition of Done. The 13% figure measures how few organizations believe they have closed that gap.

Spread across 30 products, 150,000 agents averages roughly 5,000 per product. The instinctive response is to ask how governance can

© 2026 Scrum.org | All Rights Reserved.

stretch to cover that volume. This paper asks the opposite question: how many agents can accountable humans actually govern, and what should an organization do with the answer? It works from the Developer up rather than from the enterprise down.

The Relentless Raptors showed that Scrum's accountabilities, events, and artifacts can govern a team of one human and eight agents (Hodgson & West, 2026). Most Scrum Teams have several human Developers. Once each of them governs agents, the team holds several groups of agents, and the question becomes where those groups sit in a framework that allows no sub-teams.

The Scrum community has answered a version of this question before, for human teams. Nexus lets 3–9 Scrum Teams build one product from one Product Backlog under one Product Owner, and it extends Scrum only where absolutely necessary to make that work (Schwaber, 2021). This paper explores whether the same extension holds when the teams inside the Nexus are made up of AI agents.

What Changes When AI Enters the Team

When every team member is human, the team boundary is its execution boundary. The people inside it have all the skills needed to create value each Sprint, and they decide internally who does what, when, and how. The Product Owner is accountable for maximizing the value of the product, the Scrum Master is accountable for the Scrum Team's effectiveness, and the Developers are accountable for the Sprint Backlog, for adhering to the Definition of Done, and for holding each other accountable as professionals (Schwaber & Sutherland, 2020).

When AI enters the team, each Developer can delegate execution to AI agents working within the Developer's area of expertise, against the team's Definition of Done and under the team's skill files: the written instructions that govern how agents perform each type of work. The skill files belong to the Scrum Team, and each Developer maintains the ones their own agents use. The Developer keeps the high-judgment work, particularly novel or complex work, and delegates the rest. The balance between delegated and retained work shifts as skill files mature and the Definition of Done tightens, but it never becomes all or nothing: the Developer keeps doing some of the work and directs the rest.

A front-end Developer in a product team, for example, writes React components, creates tests, and handles CSS. With AI agents, that Developer might delegate routine component work and test writing while retaining the architectural decisions and complex interaction patterns. They review the combined output of their agents, check it against the Definition of Done, and revise the skill files when the output falls short. They still contribute to front-end development, and their

© 2026 Scrum.org | All Rights Reserved.

work now also includes governing the agents that do the rest of it.

The same pattern applies outside software. A content designer on a product team might write copy, produce layouts, and generate images. With AI agents, that designer might delegate first-draft copy and image asset production while retaining the creative direction and brand judgment calls. They review the combined output, check it against the design brief and the Definition of Done, and revise the skill files when the output misses the mark.

Delegation creates a governance responsibility. The Developer ensures the combined output of their agents meets the team's quality standard, inspects it against the Definition of Done, and applies the team's governing instructions, the Definition of Done and the skill files, to their agents (Hodgson & West, 2026). The Developer's authority stops at the irreversibility boundary: the point at which a decision, once acted upon, cannot be undone without disproportionate cost. Decisions beyond that boundary, such as releasing to production, go to the Product Owner. A front-end Developer integrates a UI component into the team's Increment without asking. The same Developer does not release a breaking change to production.

Much of the execution now happens in groups of agents directed by the team's humans, so the team boundary stops being an execution boundary and becomes a governance boundary. Several Developers each directing agents leaves the team with several groups doing its work, and the Scrum Guide does not allow them to become sub-teams: within a Scrum Team there are no sub-teams or hierarchies (Schwaber & Sutherland, 2020). The groups need a legitimate place to sit, and the accountability for the Increment must stay whole. The Scrum Team remains accountable for creating a valuable, useful Increment every Sprint (Schwaber & Sutherland, 2020), whoever or whatever produced its parts.

The Agentic Nexus

Nexus provides that place. A Nexus is a group of approximately 3–9 Scrum Teams that work together to deliver a single product, with a single Product Owner managing a single Product Backlog from which all the teams work (Schwaber, 2021). Nexus adds one accountability, the Nexus Integration Team, which is accountable for producing a done Integrated Increment at least once every Sprint. It consists of the Product Owner, a Scrum Master, and the appropriate members of the Scrum Teams (Schwaber, 2021).

The **Agentic Nexus** applies that structure inside a single product team. Each human Developer's agents form a team of their own, an agent team, working from the Scrum Team's Product Backlog. The human members of the Scrum Team, its Product Owner, Scrum Master, and Developers, perform the accountability of the Nexus Integration Team across those agent teams. The humans remain one Scrum Team; what they add, once their Developers govern agents, is the integration accountability Nexus defines. Because Nexus scales Scrum by adding elements rather than changing it, the rules of Scrum continue to apply

© 2026 Scrum.org | All Rights Reserved.

to every team in the structure. The name describes the teams inside the Nexus, not who governs it: every agent team works under human accountability.

The numbers line up with the Nexus Guide. A Scrum Team is typically 10 or fewer people (Schwaber & Sutherland, 2020), which leaves room for up to 8 Developers once the Product Owner and Scrum Master are counted. A team of 3–8 Developers, each governing an agent team, produces a Nexus of 3–8 agent teams, within the approximately 3–9 teams the Nexus Guide describes (Schwaber, 2021).

The Relentless Raptors are the building block of this pattern rather than an instance of it. They are one Scrum Team in which a human Product Owner governs agents that work as the Developers, supported by an agent Scrum Master (Hodgson & West, 2026). With only one team, there is nothing for Nexus to integrate. The Agentic Nexus describes what happens when several such units, each directed by a different human, must deliver one Increment.

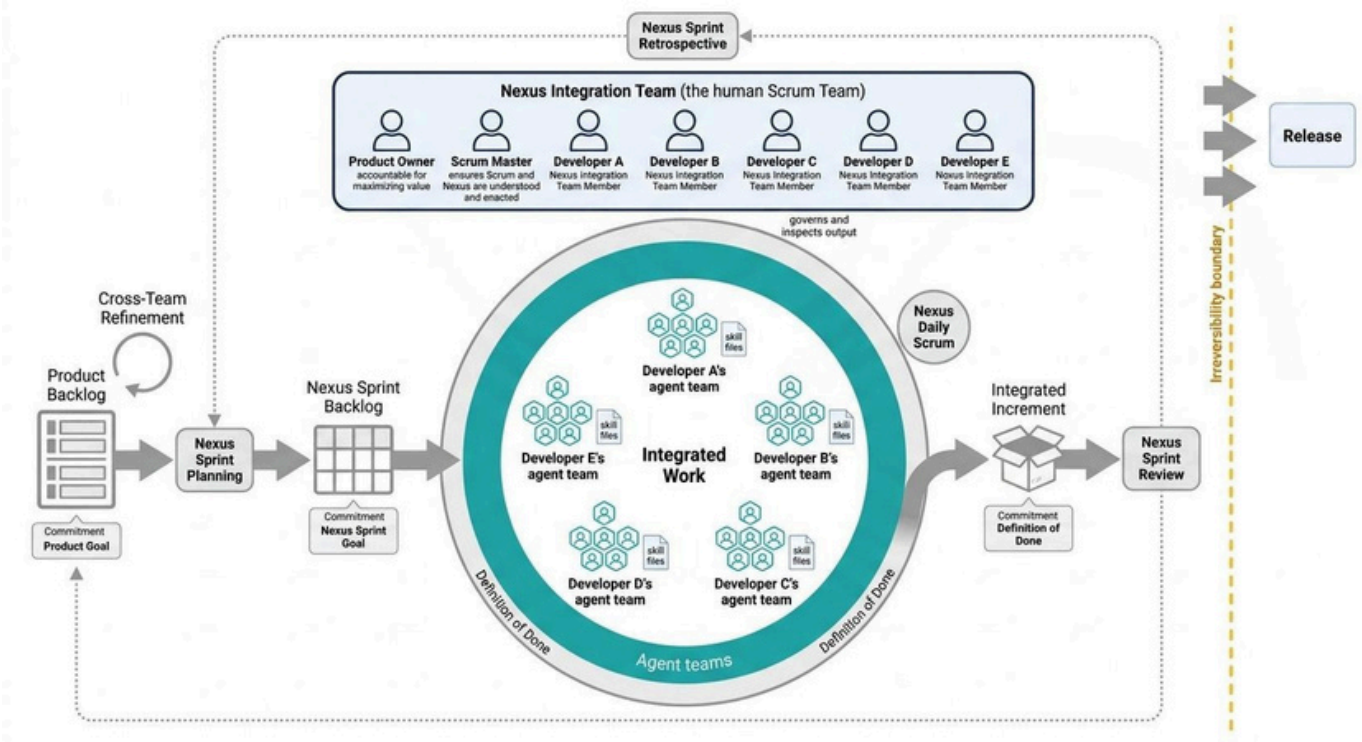

Figure 1. *The Agentic Nexus: the human members of the Scrum Team form the Nexus Integration Team, and each human Developer governs an agent team whose work is integrated into one Integrated Increment that meets the Definition of Done. Adapted from the Nexus framework diagram in The Nexus Guide (Schwaber, 2021), CC BY-SA 4.0.*

© 2026 Scrum.org | All Rights Reserved.

Nexus Element	Application in the Developer's Nexus
A single Product Backlog	Unchanged. Every agent team works from the Scrum Team's Product Backlog. As items are refined, the team makes transparent which agent team will most likely do the work, as the Nexus Guide describes for Scrum Teams (Schwaber, 2021).
A single Product Owner	Unchanged. The team's Product Owner remains accountable for maximizing value and for release decisions across all agent teams. Each Developer is the accountable human for the quality and transparency of their own agent team's output. This extends the Developer's existing accountability; the Developer does not become a Product Owner for the agents.
A Nexus Sprint Backlog	The Scrum Team's Sprint Backlog. It makes transparent the work each agent team has taken on and the dependencies between agent teams.
A single Definition of Done	Inherited from the Scrum Team and from the product level above. An agent team cannot apply a weaker Definition of Done than the team's. A Developer can apply a more stringent one to their own agent team.
Nexus events	The Scrum Team's events carry the Nexus events. Refinement identifies dependencies between agent teams before work begins. Sprint Planning coordinates work across agent teams, the Daily Scrum surfaces integration issues between them, the Sprint Review inspects the Integrated Increment, and the Sprint Retrospective inspects whether the governing instructions work across all agent teams.
Nexus Integration Team	The human members of the Scrum Team: the Product Owner, the Scrum Master, and the human Developers. They are accountable for a done Integrated Increment. Integration work takes precedence over each Developer's work with their own agent team, as Nexus Integration Team membership takes precedence over Scrum Team membership in Nexus (Schwaber, 2021). Whether this holds as agent volume grows is an open question (see Boundaries, point 4).

© 2026 Scrum.org | All Rights Reserved.

The parallel rests on a line the Scrum Guide already draws. The Product Owner is accountable for maximizing value (Schwaber & Sutherland, 2020), which makes what is worked on and when it is released their decisions. The Developers are accountable for instilling quality by adhering to the Definition of Done (Schwaber & Sutherland, 2020). The Agentic Nexus keeps that line: the Developer answers for the quality of their agent team's output and for how the work is organized within their domain, while value and release stay with the Product Owner. The Developer's accountability extends to cover agent output, and no new accountability is created.

Inside each agent team, the agents organize their own work within the structure the Developer and the Nexus provide: the ordered Product Backlog, the sequence of work in the Sprint Backlog, the Definition of Done, and the skill files. An agent team can carry its own agent Scrum Master, as the Relentless Raptors do (Hodgson & West, 2026). The Developer inspects the agent team's output, feeds findings into the team's Daily Scrum, Sprint Review, and Sprint Retrospective, and revises the skill files when the output falls short. The Developer's first commitment is to the Nexus Integration Team; once that is satisfied, they work within their own agent team, as Nexus Integration Team members do in Nexus (Schwaber, 2021).

Dochkina (2026) ran more than 25,000 tasks across groups of 4–256 agents and compared 8 coordination protocols. The best-performing protocol fixed the order in which agents worked but let each agent choose its own role. It outperformed a protocol in which a coordinating agent allocated the work by 14%, and a fully autonomous protocol by 44% (Dochkina, 2026). The Agentic Nexus mirrors that balance: the Nexus supplies order, standards, and integration, and the agents organize themselves within them. The study used synthetic tasks and AI judges and has not yet been peer reviewed, so it supports the design principle without validating the pattern.

Hackman and Vidmar (1970) found that the effect of group size on performance and member reactions depends on the type of task the group performs. Task type is also a design variable for an agent team, which makes their finding a candidate for extension to agent groups rather than a confirmed parallel.

The Scrum Master in an Agentic Nexus

The Scrum Master's accountability does not change in an Agentic Nexus; what it applies to does. The Scrum Guide makes the Scrum Master accountable for establishing Scrum as defined in the Guide and for the Scrum Team's effectiveness, including coaching team members in self-management (Schwaber & Sutherland, 2020). The Nexus Guide makes the Scrum Master in the Nexus Integration Team accountable for ensuring the Nexus framework is understood and enacted, and gives the Integration Team responsibility for coaching teams to acquire, implement, and learn practices and tools (Schwaber, 2021). Once agents do much of the work, the skill files are where the rules of Scrum act on that work, so the Scrum Master's accountability extends to the system of governing instructions.

© 2026 Scrum.org | All Rights Reserved.

That accountability separates stewardship from ownership. Each Developer owns the content of the skill files their agent team uses. The Scrum Master is accountable for the system those files form: whether the files are consistent across agent teams, whether each conforms to the Definition of Done and to the rules of Scrum, whether the revision process works, and when a change made for one agent team should become a team-wide standard. The limit is the one Evolve draws: the Scrum Master governs how the standards encoded in the instruction set change, while the protocols agents use to decompose, sequence, and execute their own work stay with the agents (Hodgson, 2026). The Scrum Master does not govern any agent team and does not take over a Developer's skill files.

In the Relentless Raptors, the Scrum Master performs the pattern analysis the agents cannot perform for themselves: tracking quality across Increments to detect skill file drift, and maintaining a constraints register that records each failure mode, the governing instruction implicated, and its resolution (Hodgson & West, 2026). In an Agentic Nexus, each human Developer can perform that analysis for their own agent team. The Scrum Master's work moves to coaching Developers to do it well, and to the analysis no single Developer can see: failure patterns that recur across agent teams, and instructions that have drifted apart. The Sprint Retrospective is the Scrum Master's main mechanism for both.

The Fractal Starts Inside the Team

The Agentic Nexus is the smallest unit of a fractal governance design, a structure in which the same pattern repeats at every scale, like a set of Russian matryoshka dolls. Each unit has the same shape: one human accountable for value, one ordered backlog, one quality standard, one integration accountability, and inspection on a regular schedule. Within a product, that human is the Product Owner and that backlog is the Product Backlog. Inside the smallest unit sits the agent team, where the accountable human is the Developer and the accountability is for quality. What changes from level to level is scope, from agent output to the team's Increment, to the product's Increment, to portfolio value. The pattern extends through the product governance hierarchy (Hodgson, 2026; Schwaber, 2021):

Level	Who Governs	What They Govern	Governance Mechanism
Agent team	Human Developer, accountable for quality	Their agent team's output	Skill files and the Definition of Done applied to the agent team; inspection of output; findings fed into the team's events

© 2026 Scrum.org | All Rights Reserved.

Level	Who Governs	What They Govern	Governance Mechanism
Scrum Team (Agentic Nexus)	Human Scrum Team as Nexus Integration Team; Product Owner accountable for value and release	Integration of all agent teams into one Increment	Product Backlog, Definition of Done, Sprint Review of the Integrated Increment, Retrospective on governing instructions
Product (Nexus of Scrum Teams)	Human Product Owner, with a Nexus Integration Team across the Scrum Teams	Integration across teams into one product Increment	Single Product Backlog, Product Goal, Nexus Sprint Review, release decisions
Portfolio	Human Portfolio Executive	Investment across products	Portfolio governance, value stream funding

Value accountability sits with one human at each level from the team upward, quality accountability for agent output sits with the Developer, and each Scrum Team remains collectively accountable for the Increment it produces (Schwaber & Sutherland, 2020). At every level, the accountable humans review evidence on a regular schedule and escalate at the irreversibility boundary.

In practice, a data engineer reviews the combined output of their agent team before it is integrated into the team's Increment. The Scrum Team and its stakeholders inspect the Integrated Increment, built by five Developers' agent teams, at the Sprint Review. In a product built by four Scrum Teams, the Nexus Sprint Review inspects the Integrated Increment of all four (Schwaber, 2021). At the portfolio level, executives review investment outcomes across their products at a quarterly review. Each review asks the same question of the evidence in front of it: are we heading in the right direction?

Empiricism at Every Level of the Pattern

The fractal governs only if empiricism operates at every level. Without transparency, inspection, and adaptation, the pattern is an organizational chart. The sprawl Gartner describes, rapid growth in agents with little governance in place, is a failure of empirical practice as much as of structural design (Gartner, 2026). Three properties make empiricism work at every level of the pattern.

© 2026 Scrum.org | All Rights Reserved.

The Definition of Done Filters Quality Before the Next Level Sees It

At the agent team level, the Definition of Done ensures each agent's output meets the quality standard before it joins the Developer's combined output. At the team level, it ensures the Integrated Increment meets the standard before the Scrum Team and stakeholders inspect it at the Sprint Review. At the product level, it ensures the integrated product Increment is usable (Schwaber & Sutherland, 2020). Each level relies on the level below to enforce the Definition of Done, which is why its precision carries more weight as agent numbers grow.

Take a Definition of Done that requires unit tests for all code. In an agent team of eight, one agent that skips the tests produces output the Developer catches before it reaches the team's Increment. A standard that omits the requirement fails in every agent that works to it, and no human can personally review every line. The cost of an imprecise standard grows in proportion to the number of agents working to it.

Inspection at Cadence Replaces Inspection at Volume

No human at any level reviews every output from the level below. The Developer does not read every line their agents produce; they inspect the agent team's combined output against the Definition of Done and the Sprint Backlog. The Product Owner does not inspect each agent team's output; the Scrum Team and stakeholders inspect the Integrated Increment at the Sprint Review. At the product level, the Nexus Sprint Review inspects the Integrated Increment of all the Scrum Teams rather than each team's work separately (Schwaber, 2021).

Inspection happens on a regular schedule, against a commitment, using evidence. The Sprint Review is the working session where the Scrum Team and stakeholders inspect the outcome of the Sprint and determine future adaptations (Schwaber & Sutherland, 2020). That rhythm holds at every level of the pattern.

Adaptation Propagates Downward Through Governing Instructions

When a Sprint Retrospective identifies a failure in a governing instruction, the revision reaches every agent governed by that instruction. A revised skill file improves the output of every agent that uses it, in every agent team that uses it. A revised Definition of Done at the product level raises the quality floor for every team's Increment.

Suppose a team's Retrospective finds that agents are producing code without adequate error handling. The Developer who governs the affected agent team updates the skill file to require error handling in every function, and in the next Sprint every agent working to that skill file handles errors. One revision changes the output of every agent that uses the instruction. The Scrum Master then checks whether the same gap exists in other agent teams, and whether the revision should become a team-wide change to a shared skill file. A constraints register kept at the level of the Agentic Nexus, recording each failure mode, the

© 2026 Scrum.org | All Rights Reserved.

agent team and governing instruction involved, and the resolution, makes those cross-team patterns visible from one Retrospective to the next (Hodgson & West, 2026).

The learning-loop framework applies at every level: single- and double-loop learning (Argyris & Schön, 1978), extended to triple-loop learning by Flood and Romm (1996) and operationalized for agentic Scrum in Hodgson and West (2026). Within an agent team, the Developer performs single-loop correction by rejecting non-conforming output and double-loop revision by updating the skill file. Triple-loop learning, questioning whether the governance approach itself is fit for purpose, belongs to the Sprint Retrospective, where the Scrum Master and the team examine whether the pattern is working across all agent teams. As agent numbers grow, the Retrospective carries more of the governance load, because it is where a single change reaches every agent team at once.

Governance Capacity Sets the Scale

The Scrum and Nexus Guides set the number of humans; neither says anything about how many agents a Developer can govern. Assume, for illustration, that each Developer governs an agent team of 5, as in Figure 1. A Scrum Team of 5 Developers then covers 25 agents. A product built by a Nexus of 6 such Scrum Teams covers 150, and a Nexus of 9 teams, each with 8 Developers, covers 360. Gartner's projection, spread across 30–50 products, implies 3,000–5,000 agents per product. At these ratios, the pattern does not reach that volume, and no evidence yet supports a higher one.

The Nexus Guide observes that adding people to a product increases complexity, dependencies, and communication pathways, and it treats scaling down, reducing the number of people working on something, as a legitimate way to deliver more value (Schwaber, 2021). Applied to agents, the principle reverses the question Gartner's forecast invites. The number of agents an organization can safely run is set by the number its accountable humans can govern, with defined authority, inspection at cadence, and governing instructions maintained as living documents (Hodgson, 2026). Agents beyond that number are ungoverned by definition, which is the sprawl Gartner warns against (Gartner, 2026).

Organizations that need more agent capacity have two legitimate routes. They can raise the governable ratio by maturing skill files and tightening the Definition of Done until Developers can safely govern larger agent teams, which each team discovers Sprint by Sprint. Or they can add governed products and value streams, each with its own Product Owner, Product Backlog, and integration accountability, rather

© 2026 Scrum.org | All Rights Reserved.

than adding oversight layers above ungoverned agents (Bittner et al., 2018). Both routes grow agent volume at the rate governance can absorb. Neither reaches 150,000 agents by deploying first and governing later.

Boundaries and Open Questions

The Agentic Nexus is a hypothesis, and seven boundaries constrain it.

1. No operational evidence exists for the Agentic Nexus. The Relentless Raptors are a single Scrum Team: one human Product Owner governing eight agents that work as the Developers, with an agent Scrum Master (Hodgson & West, 2026). With one team, there is nothing for Nexus to integrate, so the Raptors provide no evidence about integration across several humans' agent teams. No published case describes a Scrum Team whose Developers each govern an agent team. The argument for the pattern rests on the governance properties of Scrum and Nexus, not on observed practice.
2. The parallel with the Nexus Guide is structural, not validated. The Nexus Guide was written for 3–9 human Scrum Teams building a single product (Schwaber, 2021). Whether the properties that make Nexus work for human teams transfer without modification to agent teams is untested.
3. Governing capacity is constrained by factors not yet measured. Skill file maturity, Definition of Done precision, the complexity of the work, whether routine, complicated, or genuinely novel (Snowden & Boone, 2007), and the Developer's own expertise all affect how large an agent team a Developer can govern. No evidence supports a specific ratio.
4. The Nexus Integration Team's capacity is untested. The human members of the Scrum Team perform the Nexus Integration Team's accountability across agent teams, with integration taking precedence over each Developer's work with their own agents. When Developer A's agents produce output that Developer B's agents consume, the dependency is exactly what that accountability exists to manage (Schwaber, 2021). Whether the human team can hold it as agent volume grows, or whether a dedicated integration role becomes necessary, is a research question.
5. Self-management within an agent team is an open design question. When one agent's output is consumed by another within the same agent team, such as a test agent consuming a component agent's work, the agents must coordinate among themselves or the Developer must do it for them. Dochkina's (2026) results favor the first option within a fixed sequence, but they come from synthetic tasks. The same coordination problem appears within the agent team and across agent teams, which is consistent with the fractal pattern but does not prove it.
6. The human Developer's accountability inside their agent team is not yet defined. In Nexus, every team is a Scrum Team. An agent team can carry an agent Scrum Master, as the Raptors do, and it shares

© 2026 Scrum.org | All Rights Reserved.

the Nexus's single Product Owner. What the human Developer holds inside their own agent team, beyond Nexus Integration Team membership and accountability for the agent team's quality, is something neither Guide specifies and this paper does not settle.

7. The pattern's lowest unit has different properties from every unit above it. AI agents do not experience psychological safety (Edmondson, 1999), do not carry context between sessions, do not read each other's cues and adjust, and do not flag their own uncertainty as human team members do. Clear, bounded authority (Hodgson & West, 2026) and psychological safety both create conditions for effective work, but through different mechanisms. The governance pattern may transfer to agent teams; the human dynamics that make it work in human teams do not, and any implementation must account for the difference.

What This Means

For Developers

Developers who govern agents need governance competency as well as technical skill. They must manage the work delegated to AI as a backlog, enforce the Definition of Done, maintain the skill files their agent team uses, and make scope and sequencing decisions within set limits, while keeping their commitment to the Sprint Goal and to the team's accountability for a valuable Increment. That is a Scrum competency more than a technology skill.

For Scrum Masters

Scrum Masters in an Agentic Nexus need analytical competency alongside facilitation: reading failure patterns across Increments, tracing them to the governing instructions that produced them, and understanding how skill files are built, where constraints are encoded, and what makes a prohibition robust where a preference can be worked around (Hodgson & West, 2026). They coach Developers to maintain their own skill files, keep the Nexus-level constraints register, and use the Sprint Retrospective to test whether governing instructions remain consistent across agent teams. Facilitation changes too: the Daily Scrum becomes the place where Developers surface integration issues between their agent teams and adapt the Sprint Backlog accordingly, the work the Nexus Daily Scrum does across human teams (Schwaber, 2021).

For Product Owners

In a team where each Developer governs an agent team, the Product Owner still decides whether and when to release value. They do not need to inspect agent-level output to do so. The Definition of Done gives them the quality evidence release decisions depend on.

Delegation calibrated to risk becomes the operating discipline. As transparency shows the Definition of Done holding and agent teams'

© 2026 Scrum.org | All Rights Reserved.

output improving, the Product Owner can move the irreversibility boundary, delegating some release decisions to Developers (Hodgson & West, 2026). The Sprint Retrospective is where the Scrum Team tests whether that delegation is working, by examining what the combined evidence shows about quality and direction, and whether the governing instructions are still fit for purpose.

For Organizations and Executives

The shift from execution boundary to governance boundary changes headcount planning: the question becomes how many agent teams a Scrum Team can integrate, and what human competencies each governance position requires. The answer depends on skill file maturity, Definition of Done precision, and the complexity of the work (Snowden & Boone, 2007), all of which teams discover through Sprints rather than predict in advance. Agent capacity follows governance capacity, not the other way around.

Governance for AI and governance for teams are also the same design problem. The accountability structure above the team, from the Product Owner of a single product to portfolio leadership, and the Agentic Nexus inside the team follow one pattern at different scales (Hodgson, 2026). Executives who treat them as separate problems build two governance systems where one would do.

Conclusion

The framework the Scrum community built to coordinate multiple teams on a single product may also describe what happens inside a single team when AI multiplies its output. Scrum allows no sub-teams, so each Developer's agents form a team of their own inside a Nexus, and the human members of the Scrum Team take on the integration accountability Nexus defines while staying collectively accountable for the Increment. If the pattern holds at every scale, the smallest nesting doll is not the team. It is the Developer, working inside a team that owns the result.

The Scrum community did not set out to build a governance architecture for AI. It built one for teams of people. The logic of that design suggests it can extend to wherever accountable humans govern work they did not personally produce, provided the number of agents stays within what those humans can govern. Testing that claim across many teams is the next step.

© 2026 Scrum.org | All Rights Reserved.

About the Authors

Matthew Hodgson is co-founder and CEO of Zen Ex Machina, a diagnostic-first executive strategy and advisory firm operating across government and corporate Australia and the Asia-Pacific region. He is a Scrum.org Professional Scrum Trainer and has led complex organizational transformations across Australian federal agencies and corporate environments for over 20 years. He is the author of *Evolve: The Operating Model AI Demands.*

Dave West is CEO at Scrum.org, The Home of Scrum. A frequent keynote speaker and widely published author, he co-authored The Nexus Framework For Scaling Scrum and Head First Object-Oriented Analysis and Design. He led development of the Rational Unified Process at IBM/Rational, managed Ivar Jacobson Consulting for North America, and served as VP research director at Forrester Research. Before Scrum.org, he was Chief Product Officer at Tasktop, helping grow it into a VC-backed company of nearly 100.

About Scrum.org

Scrum.org, the Home of Scrum, was founded by Scrum co-creator Ken Schwaber as a mission-based organization to help people and teams solve complex problems. We do this by enabling people to apply Professional Scrum through training courses, certifications and ongoing learning all based on a common competency model.

References

Argyris, C., & Schön, D. A. (1978). *Organizational learning: A theory of action perspective.* Addison-Wesley.

Bittner, K., Kong, P., & West, D. (2018). *The Nexus framework for scaling Scrum: Continuously delivering an integrated product with multiple Scrum teams.* Addison-Wesley Professional.

Dochkina, V. (2026). *Drop the hierarchy and roles: How self-organizing LLM agents outperform designed structures* [Preprint]. arXiv:2603.28990. https://arxiv.org/abs/2603.28990

Edmondson, A. C. (1999). Psychological safety and learning behavior in work teams. Administrative Science Quarterly, 44(2), 350–383. https://doi.org/10.2307/2666999

Flood, R. L., & Romm, N. R. A. (1996). *Diversity management: Triple loop learning.* Wiley.

Gartner. (2026, April 28). *Gartner identifies six steps to manage AI agent sprawl*. Gartner, Inc. https://www.gartner.com/en/newsroom/press-releases/2026-04-28-gartner-identifies-six-steps-to-manage-artificial-intelligence-agent-sprawl

© 2026 Scrum.org | All Rights Reserved.

References Continued

Hackman, J. R., & Vidmar, N. (1970). *Effects of size and task type on group performance and member reactions. Sociometry, 33*(1), 37–54.

Hodgson, M. (2026). *Evolve: The operating model AI demands.* Zen Ex Machina.

Hodgson, M., & West, D. (2026). *Empiricism at machine speed: Scrum as a governance framework for agentic AI teams* [Whitepaper series, Papers 1–4]. Scrum.org. https://www.scrum.org/resources/empiricism-at-machine-speed

Schwaber, K., & Sutherland, J. (2020). *The Scrum guide: The definitive guide to Scrum: The rules of the game.* Scrum.org. https://scrumguides.org

Schwaber, K. (2021). *The Nexus guide: The definitive guide to scaling Scrum with Nexus.* Scrum.org. https://www.scrum.org/resources/nexus-guide

Snowden, D. J., & Boone, M. E. (2007). *A leader's framework for decision making.* Harvard Business Review, 85(11), 68–76.

© 2026 Scrum.org | All Rights Reserved.