

# Thoughtworks Technology Leaders Forum

## New York City, September 25

### Breakout Session Summaries

***A note on this document:** These summaries were created by Claude and drawn from breakout discussions held under the Chatham House Rule. In keeping with that rule, all references to individual participants and their organizations have been removed or generalized. Views represent the range of perspectives shared in each room and should not be attributed to any single company or individual.*

#### 1. Semantic models, ontologies & context metadata: what, why & how

*Using AI to fix the problem of bad "source" data that undermines AI accuracy*

The conversation opened by naming a widely shared frustration: the vocabulary around semantics, metadata management, knowledge graphs, context layers and ontology is used inconsistently across vendors and practitioners, and that confusion makes it hard for enterprises to agree on where to invest. Participants distinguished three separate questions worth separating: what these concepts actually mean, why they matter to the business, and how to implement them technically, noting that the "how" is where most disagreement lives because approaches range from lightweight markup and cataloging to full ontology frameworks built on RDF and OWL.

A recurring business anxiety surfaced repeatedly: after years of establishing clear data ownership between business and technology, leaders now hear that business units must also own semantic models and ontologies, and many admitted they have no established way to govern that. One attendee noted their organization ran into this directly on an ontology engagement, where mapping connections between systems revealed multiple conflicting versions of the same truth, showing that even a partial ontology exercise can surface valuable but uncomfortable inconsistencies.

The group converged on domain-driven design and the idea of "bounded contexts" as the most durable mental model. Rather than chasing one universal enterprise ontology, most

agreed it makes more sense to let different domains (for example HR, finance and supply chain) maintain their own definitions and relate them at the edges, echoing patterns from data mesh: keep data and its context close to the people who understand it, and let orchestration handle cross-domain translation. Several participants noted that decades of industry attempts at universal definitions (schema-style initiatives, financial-services ontology standards, and various open interoperability consortia) have struggled precisely because generic, one-size-fits-all definitions become too abstract to be usable, and that new open standards efforts should be judged on relationships and governance, not just definitions.

A security angle emerged unexpectedly: keeping ontologies scoped to bounded contexts also limits the "blast radius" of an agent's access, helping keep sensitive categories like PII contained rather than letting context leak across the enterprise. On tooling, the group debated whether graphs remain the right structure at all, agreeing that vectors, embeddings and graphs are just techniques, and the real skill is choosing the representation that matches a specific use case rather than trying to model the entire business at once.

Looking forward, participants are optimistic about AI's growing role in automatically extracting "tribal knowledge" scattered across documents and subject-matter experts' heads to bootstrap these models, while acknowledging this creates a chicken-and-egg problem: good data is needed to train AI, but AI is increasingly needed to make the data good in the first place. The suggested near-term approach is highly use-case driven: solve the most acute business problem in one bounded domain first, prove the pattern, and expand deliberately rather than attempting an enterprise-wide semantic overhaul.

## 2. Guiding the conversation: can AI reimagine (and grow) the business?

*Moving executive thinking beyond productivity gains toward new lines of business*

This discussion tackled a problem many technology leaders recognize: early AI conversations were framed around finding use cases and driving employee productivity, but leaders are struggling to push business conversations further, into genuinely reimagining what the business could become. One participant described being asked by business units what new lines of business AI should unlock, and finding that the honest answer is often "I don't know," because business leaders have spent years optimizing within existing constraints and the muscle for unconstrained strategic thinking has atrophied.

Several practical patterns emerged for surfacing bolder ideas. One is creating an explicit "sandbox" or isolated group, freed from standard budget and policy constraints, to

explore genuinely new bets; several participants said this had already produced ideas they would not have found through normal planning cycles. Another is pairing business subject-matter experts directly with engineers in the same team so that neither a purely technical nor a purely business framing dominates, rather than routing ideas through the usual "explain your requirements to IT" process. A third pattern discussed was requiring cross-functional teams (not just engineering) to compete in hackathons, which one participant said dramatically improved idea quality once product and business economics were built in alongside the technical prototype, though converting winning ideas into funded production work remains difficult.

The group repeatedly returned to a set of illustrative analogies: a well known fast-food company being reframed as fundamentally a real-estate business, and a video rental company reinventing itself as a streaming and infrastructure business, to make the point that transformative reinvention usually starts by asking what the organization is truly good at underneath its current product, rather than assuming the current product is fixed. Participants questioned whether a company's core competitive advantage (such as a strong project-management discipline or a well-organized data foundation) could be offered as a service to a completely different industry, effectively becoming a new adjacent business line.

A tension surfaced around expectations: there is enormous pressure from the top to treat AI as a hyper-efficiency lever, cutting cycle times and costs, but participants cautioned that efficiency framing can crowd out the harder, more valuable conversation about growth and reinvention. Several agreed that this kind of reimagination has to be modeled and championed from the most senior executive level to have any chance of success, since grassroots enthusiasm alone cannot overcome legacy assumptions, internal politics, and a natural human resistance to giving up familiar ways of working. The session closed without a settled answer, but with agreement that constrained execution and unconstrained imagination require fundamentally different forums, incentives and permissions to coexist.

### **3. Model security and legal implications**

#### *Navigating AI implementation amid data restrictions and vendor risk*

This roundtable focused on the tangle of data licensing restrictions, vendor dependency and legal exposure that complicates enterprise AI adoption. The starting point was that most valuable proprietary or licensed data comes with strict usage restrictions, sometimes so strict that even the software vendor providing a tool is contractually barred from viewing the underlying data, which makes troubleshooting difficult. Because data is often a core competitive differentiator, there is little appetite for openness in these industries, and some third-party data providers now explicitly prohibit their data from

being used to train AI models at all, pushing customers toward the provider's own proprietary AI tools instead. One participant noted this restriction has begun appearing even in consumer media licensing, underscoring how quickly "do not use for AI training" clauses are spreading.

Legal teams are grappling with new copyright ambiguity around AI-generated code and content, since it is often unclear who can claim ownership, and some clients now demand granular disclosure of every instance where AI was used in delivering a piece of work, creating a real administrative burden. A widely used mitigation is requiring a human reviewer in the loop before AI output is delivered, which helps satisfy the expectation that work is not purely machine generated, though participants acknowledged that actually enforcing these contractual protections against a non-compliant vendor is difficult in practice.

Organizations described divergent postures. Some have built internal platforms that route employee AI usage through a controlled gateway, giving staff access to a tiered set of frontier and open models plus internal domain-specific agents, while keeping sensitive data from ever reaching external providers. Others take a far more conservative approach, deliberately limiting AI to low-risk "acceleration" use cases, such as reducing expensive manual processing time, while avoiding creative or exploratory tasks where a hallucination could cause real damage. A shared theme was rigorous vendor management: requiring vendors to disclose every AI tool and workflow they use, restricting engagement to pre-approved models, and being willing to walk away from vendors that cannot meet those standards.

The long-term risk discussion centered on the volatility of the AI vendor landscape. Relying on an AI startup that later goes bankrupt or gets acquired creates real risk that sensitive data authorized for one purpose ends up in the hands of an unknown new owner, a scenario participants compared to a well publicized case involving a genetic-testing company's data changing hands after financial distress. If a vendor collapses or deprecates a model, a dependent business process or agent can simply stop working, and switching models across an enterprise can take six to twelve months, so continuous monitoring of alternative providers is now considered essential. The discussion closed on compute economics, comparing pre-purchased compute to airlines hedging fuel costs, and floating both "gateway" routing across models of different cost and capability, and a more speculative future where meaningful compute moves back onto employee devices.

## 4. AI team topologies and the AI operating model

*How AI is reshaping org structure, roles and workforce design*

This conversation examined how organizations are rethinking team structure and roles as AI absorbs more of the software delivery lifecycle. A central observation was that decades of increasing specialization (splitting design, product, QA and engineering into separate roles) may be reversing: as more domain knowledge gets packaged into agents, individuals are again able to work across what used to be several distinct roles, echoing how software engineers historically did everything themselves before specialization took hold. Attendees disagreed on optimal team size, with some experimenting with very small three-person teams and others finding roughly ten people, augmented by agents, to be a more natural and sustainable unit, typically a product manager, a couple of engineers, and a domain or UX specialist when needed.

Training at scale was a recurring topic. One attendee noted their organization ran engineers through a structured multi-week program to build "AI native engineering" discipline on top of coding assistants, emphasizing that the goal was not to teach people to write code, but to instill the judgment and guardrails needed to use agents safely; even so, guests noted friction as some non-engineers began doing engineering-adjacent work, occasionally without the underlying discipline. A widely echoed insight was that peer-to-peer enablement outperforms top-down mandates: identifying engineers who organically became enthusiastic early adopters and turning them into internal community leaders and coaches drove more durable adoption than formal training alone.

On organizational design, attendees described treating enablement as a temporary rotational assignment (four to six months) rather than a permanent job, pulling people from across levels and teams into a cohort, then redeploying them back into their home teams, which both refreshed the enablement function continuously and spread new practices organically. A useful rule of thumb offered was that platform engineering becomes worthwhile past roughly fifty engineers, dedicated developer enablement functions become necessary past a thousand, and formal agent-experience and DevOps-equivalent functions are needed at the largest scale.

A sharp debate centered on how CFOs are framing AI's value. Several participants pushed back on the assumption that AI adoption should translate directly into headcount reduction, arguing that in practice organizations are seeing far more value from reinvesting freed-up capacity into higher-value work, and that using AI as a pretext for layoffs actually damages an organization's ability to adapt, since surviving staff become distracted and defensive rather than focused on the transformation. One attendee noted their organization made an explicit public commitment to keep staffing levels stable while changing the skill mix, framing AI's payoff around what current people become capable of rather than how many people remain. The group agreed that the biggest structural shift is not the engineering team itself, but everything upstream in product and business strategy and downstream in operations, and that measuring value requires treating the whole value chain as the unit of analysis rather than isolated engineering metrics.

## 5. Eradication of long tail SaaS: practical or not?

### *Weighing SaaS rationalization against build-versus-buy realities*

This session opened with a mandate several participants recognized immediately: a directive from finance leadership to eliminate the "bottom 100" lowest-spend SaaS vendors, a long tail that accumulated because business units historically had broad latitude to select their own tools. The group was skeptical about how durable such initiatives tend to be, noting that similar cost-cutting mandates often lose momentum once the easy wins are exhausted, but agreed the underlying question deserves a real framework rather than a slogan.

The proposed assessment approach weighs more than raw cost: number of users, licensing spend, criticality of the workflow the tool supports, and whether viable alternatives exist in the market are all relevant, since a tool that is the only option in its category may be worth keeping even at modest scale, while overlapping tools serving the same need are strong consolidation candidates. Participants stressed this has to be evaluated case by case with periodic re-review, not solved with a single sweep.

A cautionary story anchored the build-versus-buy debate: one organization tried to replace a mainstream project management SaaS tool with an internally coded alternative, believing it was one of the simplest tools to replicate, connected it to their existing CRM and marketing systems, and reverted within about a month after the internal build fell far short on features, usability and training investment. The lesson drawn was that failures like this are rarely about coding difficulty and almost always about underestimating discovery, change management and end-user adoption. This fed a broader philosophical split in the room: some organizations treat internal coding as an absolute last resort reserved for genuinely proprietary workflows, pointing to a lean, shrinking internal technology function as evidence that discipline works, while others said their own organization's culture leans toward treating itself as a software company and defaulting to building in-house.

The conversation then turned to citizen and shadow development as an adjacent, faster-growing long tail problem: new hires arriving fluent in tools like Python and AI assistants are increasingly building their own lightweight solutions, echoing earlier waves of ungoverned Access databases and Excel-based "shadow" systems that still quietly run large parts of many enterprises today. Participants agreed this creates real value but also real risk once a tool's original builder leaves and nobody understands or maintains it, and formalizing these tools into supported systems often gets stuck behind standard governance and procurement processes.

The most provocative reframe came near the end: several participants argued the entire "SaaS or not" question is application-centric and slightly beside the point. The more

durable strategy is becoming platform-centric and data-first, ensuring every application, whether SaaS, self-built, or hybrid, draws from the same governed data source rather than requiring users to repeatedly export and re-import data between systems. In that model, SaaS tools increasingly become interchangeable front ends or "connectors" to a shared data platform, and their future value depends less on the application itself and more on whether it sits on top of trustworthy, unified data, a shift accelerated as major SaaS platforms embed third-party AI models directly into their workflows.

## 6. Headless agent use cases

### *Autonomous AI operating without continuous human prompting*

Attendees debated the definition and boundaries of "headless agents": autonomous AI systems that execute complex, multi-step tasks end to end without a human prompting each step, distinguishing them clearly from simple rule-based automation, which handles only predictable, deterministic paths. The consistent framing was that headless does not mean unsupervised; rather, it means human review is applied selectively and proportionally to risk, freeing people to focus attention on the small subset of consequential decisions rather than reviewing everything uniformly.

Root cause analysis for production incidents emerged as one of the strongest current use cases. Internal agents with full access to an organization's codebase, logs, monitoring systems and incident tickets can correlate signals across all of these sources to identify a likely cause and propose a fix, something external observability vendors generally cannot replicate because they lack that comprehensive internal access. Related to this, code review triage was discussed as a practical near-term win: classifying changes into clearly safe, clearly risky and genuinely ambiguous categories, automating the safe cases, escalating the risky ones, and using feedback loops to progressively learn which patterns can be trusted, reducing manual review load over time.

Change impact analysis using knowledge graphs and ontologies was cited as a way to shrink blast-radius notifications from thousands of potentially affected stakeholders down to the handful genuinely impacted by a given change, though attendees cautioned this only works if the underlying data quality and systems of record are well maintained; a knowledge graph built on poor data simply automates noise. On testing, guests agreed AI-generated tests are valuable but constrained by environment complexity, realistic data masking and the ongoing discipline needed to keep tests current as code evolves, and suggested agents should proactively flag when a code change likely requires a test update rather than waiting to be asked. A related idea was AI-driven environment provisioning, where agents learn API behaviors and simulate external dependencies to spin up realistic test and staging environments on demand.

Legal and negotiation workflows were discussed as an emerging, higher-risk frontier: guardrailed agents drafting contracts or negotiating within predefined limits (such as discount caps or fixed clause sets) shift the real challenge from content creation to verification and accountability. One guest warned against fully headless systems that lack any review step, since silent failures in high-stakes domains create serious accountability gaps, while noting that narrowly scoped, well-defined domains like fraud detection are a better fit for high autonomy than open-ended reasoning tasks. The session closed on a societal note, flagging the growing difficulty of verifying AI-generated media and content at scale, a challenge participants saw as directly analogous to the verification problem already emerging in code review and legal drafting, and one that enterprises will need governance frameworks for well beyond their own internal systems.

## 7. Citizen development patterns for success

### *Balancing grassroots innovation speed against enterprise risk and durability*

This session grappled with a tension every technology leader in the room recognized: employees across the business are now building their own tools and agents at a pace that outstrips traditional software governance, and the group debated whether to embrace, constrain or attempt to formalize that behavior. The consensus was that trying to shut it down entirely is neither realistic nor desirable, since citizen-built tools often surface genuine business value and legitimate process gaps faster than centralized IT could, but leaving it completely unmanaged creates real security, cost and continuity risk, particularly once a tool a team depends on outlives the person who built it.

One organization described a deliberate middle path: embedding a small number of specially trained AI engineers (drawn from a structured graduate and university-style training program) inside business functions, where the embedded engineers own connectors, pipelines and access to corporate systems while the business owns the use cases and underlying data. That organization reported meaningfully better audit outcomes than functions that let people build unsupervised, where a handful of security exposures (leaked credentials, data inadvertently exposed to the public internet) had occurred, none catastrophic but all avoidable. A recurring design pattern discussed was a rotational multi-week immersive "academy" that co-locates people from around the organization to build both technical fluency and durable cross-team relationships, which several participants felt mattered as much as the technical training itself.

Governance mechanics were debated at length: how to detect informal "shadow" builds before they become embedded in critical workflows, what criteria should trigger a formal handoff from an individual's experiment to a supported production asset, and whether a registry of self-built agents and tools (tracking ownership, cost and purpose) is needed, since several participants admitted they currently have no reliable way to answer who

owns a given agent, what it can access, or what happens to it when its owner leaves. Analogies were drawn to earlier waves of ungoverned Access databases and Excel workbooks that quietly became mission-critical without any formal review, suggesting the current wave is a familiar pattern accelerated by easier tooling rather than an entirely new problem.

An unexpected anecdote stood out: in one experiment where a set of coding agents were given a complex, loosely specified task, they spontaneously created a shared coordination document to track dependencies and warn each other about related changes, without being explicitly instructed to do so, a small but striking example of emergent collaborative behavior among autonomous agents. Participants closed by emphasizing that psychological safety and incentive design matter as much as technical controls: celebrating and rewarding people who bring their informal builds forward for review, rather than penalizing disclosure, was seen as the more effective lever for surfacing and safely scaling the best citizen-built ideas, while extracting business logic buried inside legacy systems into a shared, well-governed context layer was flagged as the necessary foundation for letting agents reliably enforce the same rules a human would.

## **8. Scaling AI adoption: from 0 to 100 km/h in six months**

*Lessons from scaling AI adoption across roughly 3,000 engineers*

The session was framed around a concrete challenge posed by one attendee: their organization has roughly three thousand engineers, and while ninety percent had tried an AI coding assistant at least once, half of that group used it barely once a month, and only a small fraction had progressed to building or sharing their own agents or multi-agent workflows. The goal posed to the group was how to move from that baseline to full, safe, sustained adoption within six to nine months without the initiative losing momentum along the way.

Two contrasting playbooks emerged. The first is a deliberately slow, foundation-first change management program, exemplified by one attendee whose organization rolled out AI across its developer base in phased waves, starting with roughly a dozen teams concentrated in one location to maximize peer learning, with dedicated embedded coaches answering questions for the first several weeks. That participant was candid that early results showed productivity temporarily dropping and visible chaos, describing this as an expected and even healthy sign that people were genuinely changing habits rather than being handed a new tool with no behavior change. The explicit goal of that program is a "zero friction" software delivery lifecycle: removing the many non-coding sources of delay, such as finding institutional knowledge or waiting on access requests, rather than

optimizing raw coding speed, since coding itself is often a modest share of an engineer's time.

The second playbook favors speed: standardizing on a small number of well-supported tools that provide built-in productivity telemetry, giving leaders per-person visibility into effort and output quickly, at the cost of the deeper behavioral transformation the slower approach aims for. One attendee described a structured, self-service five-week training curriculum that scaled from an initial cohort of ten people to several hundred, supported by internal guides, a technology radar, monthly show-and-tell sessions, and a gamified internal scorecard that assesses how "AI ready" a codebase is, based on documentation quality and testing discipline, which drove real improvement simply by making it visible.

Across both approaches, panelists agreed on several conditions for success: give people cognitive headroom rather than pushing them straight into constantly supervising many parallel agents, since burnout and fatigue set in quickly; treat technical debt reduction and legacy modernization as safe, high-value early wins that build confidence before tackling greenfield work; and invest early in full traceability, tracking which humans, agents, tools and cost centers touched every piece of delivered work, so that engineering metrics can eventually be translated into language a CFO or business sponsor will trust. A more subtle governance theme was how to handle agent identity and permissions, with most participants agreeing an agent acting on a person's behalf should be scoped no more broadly than that person's own access, ensuring full traceability and avoiding entitlement creep as autonomous agents take on more delegated work.