C H A P T E R 2

The Agentic Software Delivery Operating Model

A new organisational paradigm — not a new process map

NOVABUILD FIELD MOMENT — MOVING BEYOND TOOL ADOPTION

After the first executive reset, NovaBuild's Head of Engineering gathered her delivery leads, platform lead, security architect, product director, and CAIO sponsor. The question on the wall was simple: if agents can plan, build, test, and validate parts of the work, what should the human delivery organisation now own?

The meeting quickly exposed the deeper problem. Teams had tools, but they did not have an operating model. Nobody could say which decisions agents could make, which decisions humans had to approve, where intent became binding, or what evidence was required before production movement. NovaBuild had adopted AI faster than it had redesigned delivery.

DELIVERY LEADER DECISION — CHAPTER ORIENTATION

Leadership problem: Existing SDLC structures do not tell teams how to govern autonomous execution.
Operating decision: Establish an operating model that separates intent ownership, orchestration governance, and agent execution.
Artefact you will build: An ASDOM blueprint that clarifies layers, human positioning, decision rights, and operating cadence.

The central thesis of this book is that value from agentic AI in software delivery does not come from deploying agents into existing workflows. It comes from redesigning the operating model around the unique capabilities agents bring: sustained autonomous execution, parallel workstream management, continuous validation, and iterative correction without constant human instruction. At the same time, it must preserve human judgement at the points where judgement creates value: strategy, intent, architecture, risk, ethics, prioritisation, and accountability.

We call this the Agentic Software Delivery Operating Model, or ASDOM. It is not a process map. It is a system of roles, layers, decision rights, controls, evidence flows, and operating routines that allows an organisation to work at agentic velocity without turning software delivery into unmanaged automation. ASDOM tells the delivery leader where humans sit, where agents act, how autonomy is bounded, how validation is separated from generation, and how the system learns over time.

The operating-model problem is critical because agentic execution creates a governance paradox. The organisation wants agents to move faster than a traditional human delivery chain, but it cannot allow agents to operate as unbounded decision-makers. The answer is not to slow agents down until they resemble human teams. The answer is to create a delivery system where autonomy is granted only inside explicit boundaries, with evidence, escalation, and accountability built into the workflow.

2.1 The Three Layers of ASDOM

ASDOM has three layers. Each layer has a different purpose, a different human role, and a different failure mode. Delivery leaders should resist the temptation to blur these layers. When the layers blur, accountability blurs. When accountability blurs, autonomous execution becomes difficult to trust.

LAYER 1: The Intent Layer

- Humans define what must be built, why it matters, what constraints apply, and how success will be recognised.

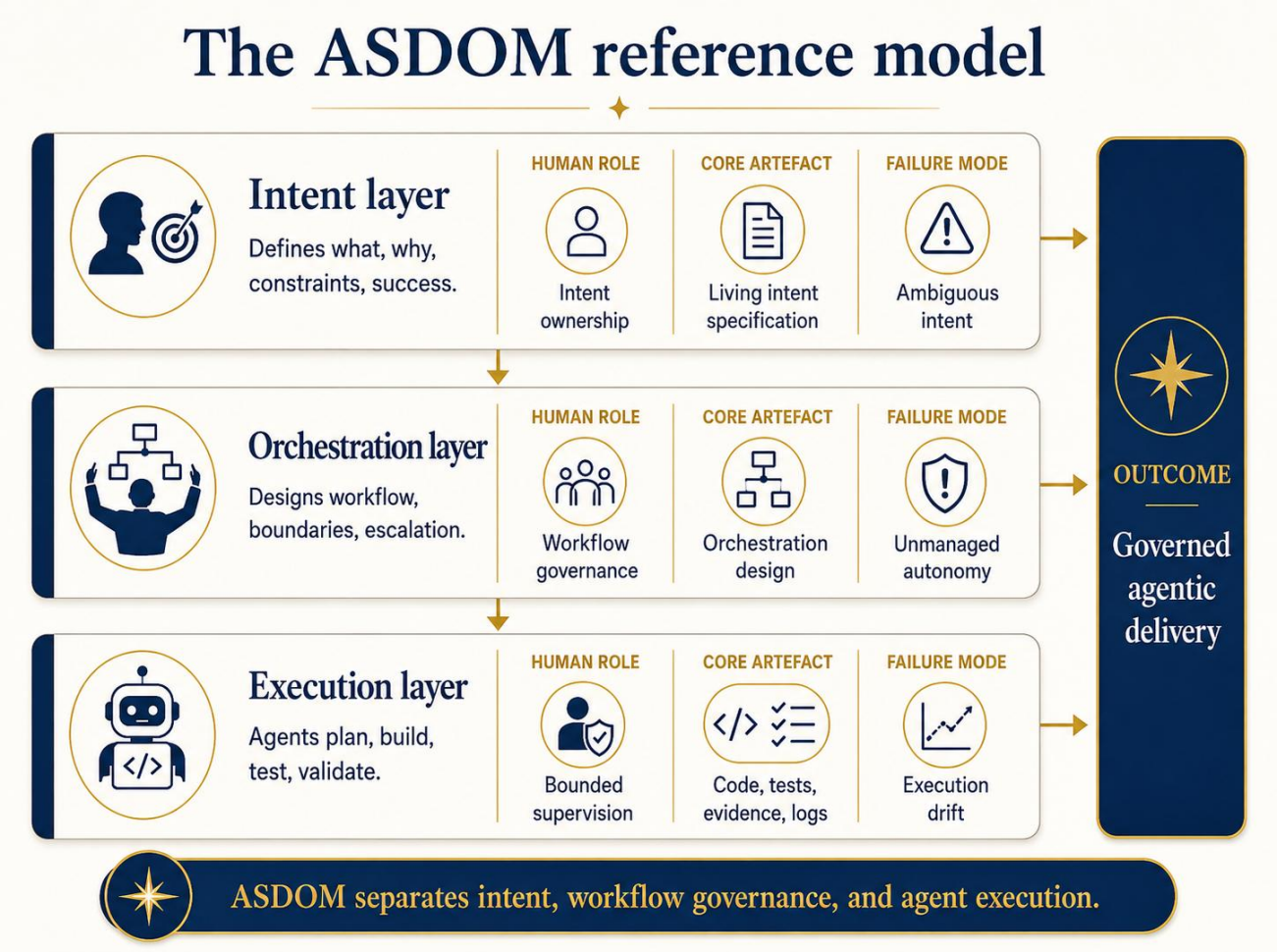
- The core artefact is the living intent specification: outcome-rich, constraint-rich, and execution-ready.
- The failure mode is ambiguous intent that agents convert into confident but misaligned execution.

LAYER 2: The Orchestration Layer

- Humans design and govern the agent workflow: planning, sequencing, validation, exception handling, and escalation.
- The core artefact is the orchestration design: which agents act, which tools they use, what evidence they produce, and when humans intervene.
- The failure mode is unmanaged autonomy: agents doing work faster than the organisation can understand or govern.

LAYER 3: The Execution Layer

- Agents plan, generate, test, validate, document, and propose deployment movement within defined boundaries.
- The core artefacts are generated code, test results, conformance evidence, logs, trace records, and deployment recommendations.
- The failure mode is execution drift: work that appears locally correct but violates intent, architecture, security, or operational constraints.



The ASDOM reference model

2.2 Human Positioning: Above, In, and On the Loop

The most important design choice in ASDOM is not which agent tool to use. It is where humans are positioned relative to the work. Delivery leaders should not treat human-in-the-loop as a generic safety phrase. It is an operating design decision. Different delivery domains, risk levels, and maturity stages require different human positions.

Above the loop means humans set direction, boundaries, metrics, and review outcomes rather than participating in every execution cycle. This is appropriate where the domain is mature, the intent is stable, validation is strong, and the blast radius is low. The human role is supervisory and strategic.

In the loop means humans approve defined decision points before agents proceed. This is appropriate for regulated changes, novel architecture decisions, production deployments, security posture changes, or material business-risk decisions. The human role is not to review everything. It is to review the points where judgement, accountability, and risk transfer require human authority.

On the loop means humans monitor continuous execution and intervene when the system crosses a boundary. This is appropriate for recurring workflows with strong instrumentation and clear escalation triggers. The human role is systems stewardship: watching patterns, improving controls, and handling exceptions rather than manually approving each step.

OPERATING PRINCIPLE — HUMAN JUDGEMENT MUST BE DESIGNED

Do not ask whether humans are 'in the loop' as a general principle. Ask where human judgement changes the quality, safety, or accountability of the outcome. Then design the loop around that point.

2.3 From Functional Silos to Outcome-Aligned Agentic Teams

Traditional delivery organisations are often built around functional silos: business analysis, development, quality assurance, security, DevOps, operations. Each function owns a stage. Handoffs connect the stages. That model made sense when the work itself was largely human-authored and sequential. In an agentic system, functional handoffs become expensive bottlenecks because agents can execute across boundaries faster than human teams can coordinate across them.

ASDOM shifts the organisational unit from function to outcome. A small group of humans co-owns an intent domain and governs the full lifecycle of delivery inside that domain. Agents execute across functions within the team's scope. Humans own the intent, orchestration, validation, and production outcome. This does not eliminate specialisation. It changes where specialisation is applied. Security, architecture, quality, and platform expertise become embedded in the operating system of the team rather than waiting at the end of the chain.

The delivery leader's redesign question is therefore not 'How do we add agents to our existing teams?' It is 'What outcome domains are stable enough to own intent, execution, validation, and improvement as one system?' The answer may be a customer onboarding domain, a claims-processing platform, a payments integration stream, or an internal developer platform capability. The point is that the team owns an outcome boundary, not a process stage.

2.4 Five Design Principles of ASDOM

ASDOM becomes useful only when it is translated into design principles that delivery leaders can apply repeatedly. The following principles should guide every team redesign, pilot selection, platform investment, and governance decision.

- Intent before execution: agents should not begin meaningful autonomous work until the intent is sufficiently complete, constrained, and verifiable.
- Autonomy within boundaries: agents should be given freedom inside explicit scope, authority, quality, and escalation boundaries, not blanket permission to optimise locally.
- Independent validation: the same system that generates output should not be the only system that certifies it as fit for production.
- Human judgement at explicit decision points: human review should be designed around value and risk, not anxiety or habit.
- Measurement by system outcomes: success should be measured through lead time, conformance, intervention rate, cost per outcome, and business value, not code volume or AI usage.

DESIGN TEST: Is this ASDOM-aligned?

- Can the team identify the intent source of truth before execution begins?
- Can the team explain what agents can decide autonomously and what requires human approval?
- Can validation be performed independently of generation?
- Can the delivery leader see the evidence trail for conformance, risk, and deployment readiness?

2.5 Decision Rights and Accountability

The hardest part of agentic delivery is not generating output. It is deciding who has the right to commit intent, alter boundaries, approve exceptions, accept risk, and move software into production. In traditional delivery, many of these rights are embedded informally in role expectations. In agentic delivery, informality becomes dangerous because agents will act on what is explicit, not what the organisation assumed.

A delivery leader should assign decision rights at the level of the delivery system. Intent commitment should sit with the accountable product or domain owner, supported by an Intent Engineer. Architecture boundary changes should sit with the Constraint Architect or equivalent authority. Execution planning may be agent-generated, but the orchestration pattern belongs to the Agent Orchestrator. Conformance evidence should be produced independently. Production movement should be governed by risk classification and bounded autonomy rules.

This accountability model is not bureaucracy. It is the minimum scaffolding required for speed. Teams move faster when they know who can decide, what evidence is required, and when escalation is mandatory. Agents move faster when the boundaries are explicit. Leaders move faster when the evidence is visible.

NOVABUILD FIELD MOMENT — DECISION

NovaBuild's Head of Engineering introduced one non-negotiable rule: no agentic workflow would enter execution unless four questions had named answers. Who owns the intent? Who owns the orchestration design? Who owns independent validation? Who accepts production risk?

The rule slowed the first two weeks of the pilot. It prevented six months of uncontrolled automation debt.

2.6 The ASDOM Operating Cadence

An operating model is not real until it has a cadence. ASDOM requires a rhythm through which intent, execution, validation, exceptions, and improvement are reviewed. Without cadence, the model becomes a diagram. With cadence, it becomes a management system.

A practical cadence starts with intent review. The team examines whether the upcoming work has clear outcomes, constraints, interfaces, acceptance criteria, and exclusion boundaries. The next step is orchestration planning: deciding how agents will decompose and execute the work, where validation will occur, and which decisions require human intervention. Execution then proceeds inside the agreed boundaries, with traces, logs, and evidence captured as the work progresses.

The most important recurring meeting is not a status meeting. It is the exception review. Delivery leaders should ask where agents required unplanned human intervention, where intent was insufficient, where validation failed, and where boundaries were too tight or too loose. Each exception is a learning event for the operating model. The goal is not merely to fix the current work item. The goal is to improve the specification patterns, orchestration templates, validation harnesses, and autonomy rules so the next cycle is stronger.

- Intent review: is the specification execution-ready?
- Orchestration review: is the agent workflow appropriate for the risk and complexity of the work?
- Conformance review: does generated output satisfy intent, constraints, and interface contracts?
- Exception review: where did the system need human intervention outside planned decision points?
- Learning review: what should be added to templates, controls, patterns, or platform capabilities?

2.7 The Transition Pattern: From Pilot to Operating Model

ASDOM should not be rolled out as a grand organisational reorganisation. It should be introduced through bounded domains where the organisation can learn safely. A sensible transition begins with one domain, one team, one intent pattern, one agent workflow, and one validation harness. The early goal is not enterprise scale. It is operating-model proof.

The pilot should deliberately include enough complexity to be meaningful but enough containment to be safe. A trivial pilot teaches the wrong lesson because it makes agentic delivery appear easier than it is. A high-blast-radius production domain creates unnecessary risk. The best early domains are representative, bounded, measurable, and led by people willing to operate differently.

As maturity increases, ASDOM expands through patterns rather than announcements. Intent templates become reusable. Orchestration designs become standardised. Validation harnesses become stronger. Autonomy boundaries become more permissive where evidence supports them. The delivery organisation gradually shifts from isolated agent usage to a governed delivery system.

OPERATING PRINCIPLE — SCALE THE OPERATING MODEL BEFORE AGENTS

Do not scale agents before scaling the operating model. The sequence is intent discipline, orchestration design, independent validation, bounded autonomy, then scale.

2.8 Platform Implications of ASDOM

ASDOM cannot be sustained through meetings alone. It requires platform support. The delivery organisation needs a place where living specifications are authored, versioned, reviewed, and connected to execution. It needs orchestration infrastructure that can coordinate planner, executor, validator, and specialist agents. It needs conformance checks that run continuously rather than only during manual review. It needs observability that captures agent decisions, tool calls, validation results, exceptions, and human interventions.

This platform layer should not be confused with buying a single agent tool. The platform is the internal delivery infrastructure that makes agentic work governable. Some components may be vendor-supplied, some may be built internally, and some may extend existing DevOps and developer-platform capabilities. What matters is that the delivery leader treats the platform as operating infrastructure, not as a collection of disconnected AI experiments.

The strongest early platform investments are usually not the most glamorous ones. Specification versioning, conformance evidence, traceability, policy-as-code integration, secure tool access, and exception routing create more enterprise value than another coding assistant rollout. These capabilities make autonomy safer. Without them, scale simply increases the volume of unmanaged output.

2.9 Anti-Patterns That Signal ASDOM Is Being Misapplied

Operating models fail when organisations adopt the language without changing the work. The first anti-pattern is orchestration theatre: humans continue to make every meaningful decision, but the work is described as agentic because tools are involved. The second is autonomy theatre: agents are granted broad freedom, but the evidence and escalation mechanisms are too weak to support trust. The third is governance theatre: boundaries are written down but not technically enforced or operationally reviewed.

A delivery leader should watch for these patterns early. If teams spend more time reviewing AI output than they previously spent creating the work, orchestration has not been designed well. If agents repeatedly ask for clarification during execution, intent is not ready. If security and architecture teams remain late-stage gatekeepers, governance has not been embedded. If leadership can see adoption metrics but not conformance or intervention metrics, measurement is still trapped in the augmentation era.

EXECUTION ARTEFACT — ASDOM QUALITY GATE

A team is not ASDOM-aligned because it uses agents. It is ASDOM-aligned when intent is versioned, autonomy boundaries are explicit, validation is independent, exceptions create learning, and delivery metrics show system-level improvement.

2.10 The First 90 Days of ASDOM Adoption

ASDOM should not begin as an enterprise-wide reorganisation. The first ninety days should establish credibility through a bounded operating-model pilot. The delivery leader should choose one domain with enough delivery complexity to be meaningful, but enough containment to prevent a failure from becoming a production crisis. The purpose is to prove the operating model, not to automate the organisation in one move.

The first month is about clarity. The team defines the domain, names the intent owner, identifies the orchestration lead, chooses the validation approach, and captures the current baseline. It also documents which decisions agents may make, which decisions require human approval, and which decisions are outside the pilot boundary. If this feels slow compared with simply turning on more tools, that is precisely the point. ASDOM starts by designing the control environment before expanding execution.

The second month is about execution under supervision. Agents plan and execute bounded work, but humans remain deliberately in the loop at the most important checkpoints: intent commitment, execution-plan approval, conformance review, and release readiness. The delivery leader should treat every exception as operating-model data. If agents repeatedly escalate the same type of decision, the issue may be weak intent, unclear constraints, or a poorly designed autonomy boundary.

The third month is about hardening. The team should decide which approvals can move from in-the-loop to on-the-loop, which controls must remain human-owned, and which parts of the operating cadence need to be standardised before scaling. A successful ninety-day cycle does not merely ship a feature faster. It produces reusable intent patterns, clearer decision rights, a working validation rhythm, and a credible view of what the next domain will require.

DELIVERY LEADER DECISION — READY TO SCALE ASDOM?

You can name the owner of intent, orchestration, validation, platform, and production accountability.
The pilot has measured baseline and post-pilot delivery signals.
Exceptions have been analysed and converted into improved patterns or controls.
The team can explain which autonomy boundaries expanded and why.
The sponsor understands that scale means repeating the operating model, not multiplying tools.

2.11 How ASDOM Changes Management Work

ASDOM does not only change engineering work. It changes management work. A delivery leader operating in the traditional model spends much of their attention on throughput, capacity, dependencies, escalations, and delivery dates. Those concerns do not disappear, but the management surface changes. The leader must now manage the quality of intent, the design of autonomy boundaries, the health of validation systems, and the rate at which exceptions are converted into better operating patterns.

This is a different kind of management. It is less about asking whether teams are busy and more about asking whether the delivery system is learning. A high-performing ASDOM team should need fewer clarifications over time, not simply produce more artefacts. Its agents should escalate more precisely, not less often at any cost. Its validation harness should catch drift earlier. Its operating reviews should produce better patterns, not only better status reports.

DELIVERY LEADER DECISION — MANAGEMENT SHIFT

Traditional delivery management asks: are teams progressing against plan?
ASDOM management asks: is the system becoming better at converting intent into governed execution?
Traditional delivery management optimises utilisation.
ASDOM management optimises judgement placement, autonomy boundaries, conformance evidence, and reusable patterns.

2.12 Where ASDOM Should Not Be Applied First

Delivery leaders also need restraint. ASDOM should not be introduced first in the organisation's most politically sensitive, least documented, or highest-blast-radius domain. A poor first domain can make a sound operating model look reckless. The right early domain is meaningful enough to test the model, but contained enough to permit learning. It should have a committed team, known stakeholders, accessible telemetry, and enough architectural boundary clarity for agentic execution to be governed.

Avoid domains where requirements are verbal, ownership is fragmented, and production risk is poorly understood. Avoid domains where the team is already in crisis and wants agents to compensate for broken discipline. Avoid domains where success depends on undocumented tribal knowledge that cannot be translated into intent or constraints. In these environments, ASDOM will not fix the operating model; it will reveal how weak the current operating model already is.

The best first domain is not necessarily the easiest one. It is the one where learning will transfer. A small internal tooling workflow may be safe, but it may not teach the organisation how to manage real enterprise constraints. A bounded customer-facing capability with clear interfaces, known controls, and manageable production risk may produce more valuable learning. The delivery leader's job is to select for transferable learning, not for cosmetic success.

2.13 The ASDOM Scale Decision

At the end of the first ASDOM cycle, the delivery leader should not ask, 'Did the agentic pilot work?' That question is too vague. The scale decision should ask whether the operating model produced evidence that can be trusted: clearer intent, fewer unplanned escalations, faster execution within boundaries, independent validation, and a better understanding of which human decisions must remain protected.

- Scale when the team can explain its decision rights and evidence trail without heroic interpretation.
- Harden when the model works but still depends on too much expert intervention.
- Pause when intent quality, platform maturity, or governance clarity is too weak for expanded autonomy.
- Stop or redesign when the domain proves unsuitable and the learning does not transfer.

This discipline prevents ASDOM from becoming another transformation label. The model earns expansion only when it proves that delivery can become faster because the operating system is better, not because a small team worked harder to make a pilot look successful.

2.14 Role-Level Operating Examples

ASDOM becomes clearer when leaders examine how a familiar role behaves differently inside the model. A product leader no longer hands a backlog to engineering and waits for interpretation. They own outcome clarity and participate in shaping execution-ready intent. An architect no longer produces a static design for developers to follow. They define the permissible solution space and identify where agents require explicit boundaries. A quality leader no longer measures assurance mainly through test coverage. They design the conformance harness that determines whether agent-generated output expresses the specification.

The engineering manager's role also changes. In the traditional model, the manager often focuses on capacity, sprint flow, impediments, and coordination across functional teams. In ASDOM, the manager watches a different set of signals: quality of intent entering execution, frequency of agent escalation, review burden on senior engineers, conformance failure patterns, and whether exceptions are being converted into reusable workflow improvements. This is still management, but it is management of a delivery system that contains autonomous execution.

The platform leader becomes more central. Their platform is no longer only a developer experience layer. It becomes the operating infrastructure through which specifications are versioned, agents are orchestrated, validation evidence is captured, and autonomy boundaries are enforced. If the platform does not support ASDOM, each team will invent its own version of agentic delivery, and the enterprise will create fragmentation faster than it creates capability.

DELIVERY LEADER DECISION — ROLE TRANSLATION

Product becomes outcome and intent ownership.
Architecture becomes constraint and boundary ownership.
Engineering becomes orchestration and integration ownership.
Quality becomes conformance and evidence ownership.
Platform becomes the operating infrastructure for safe autonomy.

2.15 How Exception Loops Improve the Model

Every agentic workflow will produce exceptions. The question is whether the organisation treats those exceptions as interruptions or as learning. In immature implementations, exceptions are handled one by one. A human clarifies the ambiguity, approves the edge case, or fixes the output. The immediate problem is resolved, but the operating model does not improve. The next cycle repeats the same pattern.

In ASDOM, exceptions feed the model. If a planner repeatedly escalates because requirements omit data-retention constraints, the intent template must change. If a validator repeatedly fails outputs because interface assumptions are unclear, the architecture pattern must change. If a security validator blocks generated dependencies, the execution environment and approved dependency rules must change. The exception is not only a local failure. It is evidence about the system.

This is where delivery leaders should spend disproportionate attention in the early stages. A clean pilot with no exceptions may simply be too easy. A useful pilot produces enough exception data to reveal where intent, constraints, validation, platform, or decision rights require hardening. The leader's task is to make sure those lessons become institutional patterns rather than tribal knowledge.

2.16 The Operating Review That Keeps ASDOM Honest

A monthly ASDOM operating review should not become another status meeting. It should focus on whether the delivery system is becoming more capable. The review should examine intent quality, autonomy boundaries, escalation patterns, conformance results, review burden, production evidence, and the health of the platform layer. The delivery leader should ask where the system learned, where it merely worked around a problem, and where the next expansion of autonomy is justified by evidence.

- Which decision types still require too much senior human intervention?
- Which validation failures indicate weak intent rather than weak execution?
- Which controls can safely move from in-the-loop to on-the-loop?
- Which team-created patterns should become enterprise standards?
- Which domains are not yet mature enough for ASDOM expansion?

This review is the mechanism that prevents ASDOM from becoming theatre. It links the operating model to evidence and makes expansion conditional on maturity, not enthusiasm.

The first sign of sustainability is behavioural, not technical. Teams begin asking for better intent before they ask for more agent capacity. Architects define constraint boundaries earlier. Quality leaders look for conformance signals instead of waiting for test reports. Platform teams improve the orchestration path because they can see where humans are repeatedly intervening. When those behaviours appear without executive prompting, ASDOM has moved from programme language into delivery muscle.

This is why the operating model and the platform cannot be separated. ASDOM defines how decisions should be made. The platform makes those decisions executable. The leader's task is to keep both moving together so the organisation does not end up with a strong model on paper and a weak delivery system in practice.

The third condition is often underestimated. Delivery leaders can announce a new operating model, but teams will follow the path of least resistance. If the easiest path is still to write a vague story, ask an agent for output, and push review burden downstream, the old model will survive under new language. If the platform makes intent capture, orchestration, conformance validation, and exception logging natural parts of the workflow, ASDOM becomes easier to adopt and harder to bypass.

ASDOM becomes sustainable only when the organisation stops treating it as a special programme and starts treating it as the normal way selected delivery domains operate. That requires three conditions to be visible before scale. First, teams must understand the decision rights well enough to act without asking senior leaders to interpret the model every week. Second, the evidence trail must be strong enough that risk, security, and operations can trust the workflow without adding manual control layers. Third, the platform must make the right behaviour easier than the old behaviour.

2.17 The Conditions for Sustainable ASDOM Adoption

CHAPTER LEADERSHIP MOVES

- ASDOM is an operating model, not a process map or tool architecture.
- The three layers — Intent, Orchestration, and Execution — separate human judgement, workflow governance, and agent action.
- Human positioning must be designed: above, in, or on the loop depending on risk, maturity, and evidence quality.
- Outcome-aligned agentic teams replace functional handoff chains as the primary organisational unit.
- ASDOM becomes real through decision rights, operating cadence, and continuous learning from exceptions.

BRIDGE TO THE NEXT CHAPTER

Chapter 3 moves upstream to the most important asset in ASDOM: intent. If autonomous execution depends on the quality of the specification, then intent becomes the new unit of delivery.