

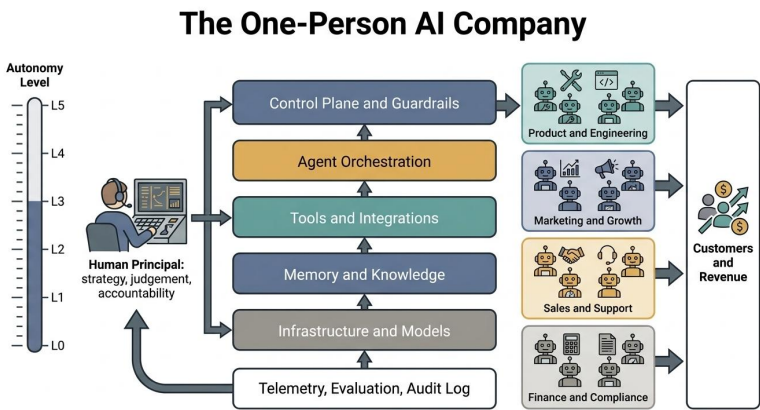
One-Person AI Company

Principles, Workflows, Architectures, and Practices
for Autonomous Solo Enterprises

Ngo Trung Kien, PhD

The Research Ledger · Volume 3

First edition



Contents

Preface	xi
How to Use This Book	xv
Acknowledgements	xix
Contents	xxi
List of Figures	xxvii
List of Tables	xxix
List of Listings	xxxi
Abbreviations and Notation	xxxiii

Part I Foundations and Business Architecture	1
---	----------

1 The Paradigm of the Autonomous Solo Enterprise	3
1.1 Why firms have the size they do	5
1.2 What actually changed: the delegation evidence	8
1.3 The one-person economy in numbers	12
1.4 Defining the one-person AI company	14
1.5 Workflows and agents, and why the distinction is load-bearing	15
1.6 Levels of enterprise autonomy	17
1.7 The delegation calculus and the interior optimum	23
1.8 The three scarce resources: verification, authority, attention	26
1.9 Revenue per operator hour, and how to read this book	28
Review Questions	31
2 The Agentic Enterprise Stack: Architecture, Memory, and Orchestration	33
2.1 The six layers and why this decomposition	35
2.2 The data and memory layer	38
2.3 Context engineering as an engineering discipline	41
2.4 The tool and action layer	44
2.5 Orchestration: workflows, agents, and the cost of coordination	47
2.6 Reliability engineering of agent loops	50
2.7 The model layer: tiers, serving mechanics, and routing economics	54
2.8 The assurance layer and the control plane in outline	58
2.9 The agentic organisation chart	59
2.10 Build, buy, or abstain	60
Review Questions	63

Part II Autonomous Operations: Product, Growth, and Customers 65**3 Autonomous Product Development and Engineering 67**

3.1	What agentic coding can and cannot do now	70
3.2	Opportunity discovery as an automated function	72
3.3	Specification-first development	73
3.4	The coding agent loop	75
3.5	The verification stack	79
3.6	Continuous delivery for one person	83
3.7	Synthetic evaluation and real telemetry	86
3.8	Maintainability and the code you did not write	87
3.9	Delivery metrics for a one-person engineering function	89
	Review Questions	91

4 Automated Marketing, Content, and Growth Engines 93

4.1	What the growth function has to produce	95
4.2	Demand discovery and competitive monitoring as agent workloads	99
4.3	The brand knowledge base	101
4.4	The production pipeline and its human gate	103
4.5	The shift to generative search	105
4.6	Disclosure, authenticity, and platform policy	107
4.7	Personalisation at scale and its limits	110
4.8	Measurement: attribution, incrementality, and small samples	111
4.9	Rate limits, spam classification, and operational hygiene	113
4.10	The interface to unit economics	114
	Review Questions	117

5 Autonomous Sales, Customer Acquisition, and Support 119

5.1	The revenue function as a queue of decisions	121
5.2	Lead qualification and routing	123
5.3	Deflection versus resolution	124
5.4	The tiered cascade and its economics	125
5.5	Authority limits in customer-facing agents	129
5.6	Escalation design	133
5.7	Disclosure and consumer trust	135
5.8	Failure modes with legal consequence	136
5.9	Instrumentation	139
5.10	Sizing the human residue	140
	Review Questions	142

Part III Economics, Law, and Security 145**6 Financial Operations, Unit Economics, and Cost Governance 147**

6.1	Choosing the cost object	149
6.2	The delivered cost model and the rework tax	151
6.3	The price of capability over time	154
6.4	Pricing the output	155
6.5	Retention, acquisition, and composite health	157
6.6	The financial control architecture	160
6.7	The budget governor	164
6.8	Cost observability and attribution	169
6.9	Bookkeeping, invoicing, and reconciliation as agent workloads	171
6.10	Reserves, runway, and the financial failure modes of autonomy	172

Review Questions	175
7 The Legal, Regulatory, and Intellectual-Property Perimeter	177
7.1 The perimeter as an engineering artefact	179
7.2 Authorship and copyright in AI-assisted output	182
7.3 Inputs and provenance	185
7.4 Contracts formed by machines	187
7.5 The EU AI Act as the reference regime	189
7.6 Data protection	191
7.7 Consumer protection and advertising	194
7.8 Liability allocation and insurance	196
7.9 The evidence pack	198
7.10 A proportionate compliance programme for one person	201
Review Questions	203
8 Security Architecture and Threat Modelling for the Solo Enterprise	205
8.1 Why the solo operator is a distinctive security case	207
8.2 The threat model: assets, boundaries, adversaries	208
8.3 The injection family	212
8.4 Tool and supply-chain compromise	214
8.5 Excessive agency and the confused deputy	215
8.6 Memory and knowledge-base poisoning	218
8.7 Design-level defences and their limits	220
8.8 Containment	224
8.9 Detection, audit, and forensics	227
8.10 A proportionate security programme	228
Review Questions	231
Part IV Governance, Resilience, and Practice	233
9 Human-in-the-Loop Governance and Control Planes	235
9.1 What cannot be delegated	238
9.2 What the human-factors evidence actually shows	239
9.3 Six oversight modes and how to route into them	241
9.4 The delegation rule and the honest estimation of p_e	246
9.5 Capacity, queues, and the binding constraint of the enterprise	248
9.6 Span of control: what is known and what is not	252
9.7 Designing the approval artefact	254
9.8 Alert fatigue and escalation hygiene	257
9.9 The control plane: what to build first	259
9.10 The operator's operating rhythm	261
Review Questions	263
10 Operational Resilience, Continuity, and Continuous Assurance	267
10.1 The failure taxonomy of a one-person operation	269
10.2 Provider dependence and the model abstraction layer	271
10.3 The degradation ladder	277
10.4 Availability arithmetic and the limits of redundancy	278
10.5 Deprecation, silent updates, and drift	281
10.6 The evaluation harness and the release gate	282
10.7 Observability: what to instrument first	288
10.8 Error budgets and release discipline for one person	289
10.9 Incident response with a single operator	291
10.10 Continuity, recovery objectives, and the key-person problem	295

Review Questions	298
11 Case Studies in Autonomous Solo Enterprises	301
11.1 Methodology: what these cases are and are not	303
11.2 Case A: a research and publishing micro-enterprise	305
11.3 Case B: a multi-tenant media and marketing consultancy	310
11.4 Case C: a developer-tools and API business	317
11.5 Cross-case analysis	324
11.6 The benchmark frame	328
11.7 What the cases do not show	330
Review Questions	332
12 Capstone: Building and Deploying Your One-Person AI Company	335
12.1 The capstone brief	336
12.2 Weeks 1–2: opportunity selection and the economic model	341
12.3 Weeks 2–3: architecture and the tool inventory	343
12.4 Weeks 3–5: the control plane first	344
12.5 Weeks 4–7: the first autonomous workflow	347
12.6 Weeks 6–8: the evaluation harness and the reliability evidence	348
12.7 Weeks 7–9: handing a function to an agent team	351
12.8 Weeks 9–10: the legal and tax pack as files	353
12.9 Weeks 10–11: the resilience drill	354
12.10 Weeks 11–12: launch, pricing and measurement	355
12.11 Outcomes and honesty	358
12.12 The maturity model and the next twelve months	360
Review Questions	363
A Reference Architecture Blueprint and Inventory Templates	365
A.1 The authority map	365
A.2 Tool and credential inventory	365
A.3 Architecture decision record	365
A.4 Agent specification template	367
A.5 Minimum viable control plane	368
B Control Catalogue with Standards Mapping	369
B.1 How to use the catalogue	369
B.2 Catalogue	369
B.3 Coverage against published risk catalogues	372
C Financial and Reliability Model Reference	375
C.1 Delegation and autonomy	375
C.2 Capacity and queueing	375
C.3 Reliability	376
C.4 Unit economics	376
C.5 Parameter values used in the figures	377
C.6 A minimal spreadsheet layout	377
D Capstone Deliverables and Assessment Rubric	381
D.1 Deliverable specifications	381
D.2 Milestone gates	381
D.3 Assessment rubric	381
D.4 Automatic failure conditions	381
D.5 Twelve-week schedule with assessment points	384

E	Guidance for Selected Review Questions	385
E.1	Method by chapter	385
E.2	Common errors across the book	387
	Glossary	389
	Bibliography	393
	Index	421

The Paradigm of the Autonomous Solo Enterprise

A firm is not a building or a headcount; it is a bundle of decisions that someone must own. Automation changes who executes the decisions, never who owns them.

— Operating premise of the one-person AI company

Learning Objectives

- Derive the classical explanation of firm size from transaction and coordination costs, and state the conditions under which falling coordination cost contracts rather than expands the optimal firm.
- Report the principal experimental evidence on delegated knowledge work with its effect sizes, settings and external-validity limits.
- Specify the one-person AI company as a governance form, distinguishing it from a solo consultant with a chatbot and from an unsupervised agent.
- Classify a system as a workflow or an agent using the control-flow criterion, and justify why the distinction determines the control design.
- Classify an operation on the L0–L5 enterprise autonomy ladder by who holds the control loop, the blast radius and the governance obligation.
- Compute the total cost of operating at a given autonomy level and locate the interior cost-minimising level.
- Compute revenue per operator hour and the maximum feasible review fraction implied by a revenue target.
- Justify verification, authority and attention as the binding constraints of the form, and trace each to its later treatment in this book.

The interesting question about a one-person company is not whether software can write marketing copy or fix a failing build. It is why the firm has the size it has. Economics has a serviceable answer, now almost ninety years old: an organisation grows until the cost of arranging one more transaction inside it equals the cost of arranging that transaction across a market [43]. Everything a firm does that is not production is coordination: specifying work, finding someone to do it, agreeing terms, monitoring performance, correcting errors, and bearing the consequences of the errors that escape correction. Headcount is the visible residue of that coordination

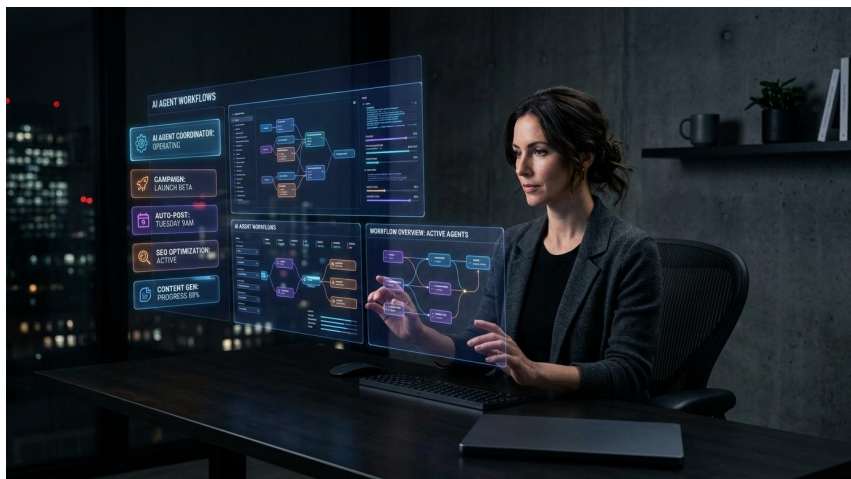


Figure 1.1: The Autonomous Solo Enterprise Paradigm: an operator directing a coordinated fleet of autonomous software agents across product, growth, and governance workflows from a single unified cockpit.

burden. If the burden falls sharply, the residue should fall with it — and the smallest viable unit of enterprise should get smaller.

That is the claim this book examines and, in a bounded form, defends. The bounded form matters. Coordination cost does not vanish under delegation to machines; it changes category. What used to be the cost of instructing and supervising colleagues becomes the cost of specifying, constraining, verifying and auditing autonomous execution. A recent formal treatment makes this explicit: agentic systems collapse integration cost while verification cost scales with task throughput rather than with the number of interacting parties [117]. The managerial literature reaches the same conclusion from the opposite direction, arguing that residual coordination, monitoring and accountability costs persist even as the marginal cost of execution collapses, so firms do not dissolve into atoms [34, 133]. The one-person AI company is therefore not the limit case of a firm shrinking to nothing. It is a specific architecture: a single accountable human operating a governed execution layer whose capacity is bounded not by hours of labour but by hours of trustworthy verification.

This chapter builds that architecture’s foundation in three movements. The first is theoretical: why firms have boundaries at all, and what a fall in coordination cost does to them. The second is empirical: what randomised and field evidence actually shows about delegating knowledge work to language models, reported with effect sizes, settings and the caveats that make several of the headline figures non-transferable. The third is definitional and quantitative: a precise statement of the form, a six-level autonomy ladder, a cost model that locates the optimal level of autonomy in the interior rather than at the maximum, and a productivity metric — revenue per operator hour — that the rest of the book is organised around improving.

Fig. 1.2 is the organising schema for the entire book, and it is worth studying before reading further, because every later chapter is an elaboration of one of its

parts.

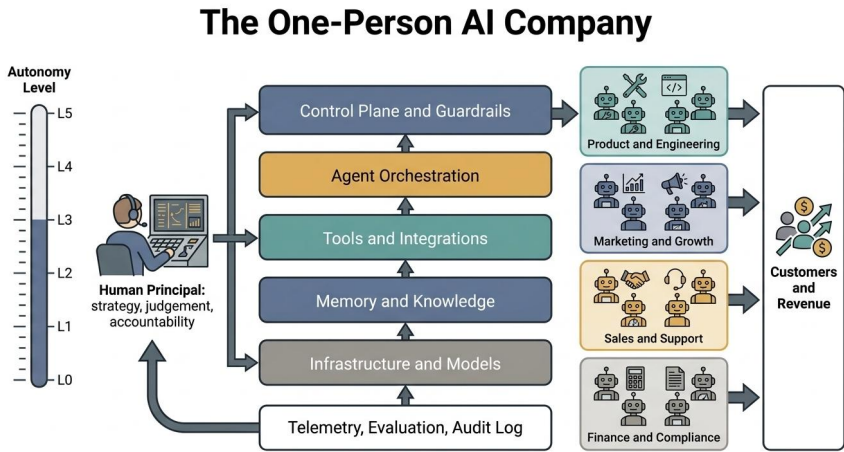


Figure 1.2: The organising schema of this book. A single *human principal* (left) holds strategy, judgement and accountability, and works through a control-plane console rather than through direct production. The centre shows the five stack layers, bottom to top: *infrastructure and models* (compute and inference, Chap. 2 and Chap. 6); *memory and knowledge* (durable state and retrieval, Chap. 2); *tools and integrations* (the effector surface, Chap. 2 and Chap. 3); and *agent orchestration* (task decomposition and scheduling, Chap. 2 and Chap. 3); and *control plane and guardrails* (policy, approval gates, budget and kill switches, Chap. 9). The reference architecture of Chap. 2 resolves this last band into two layers, separating *assurance* — the policy engine, evaluation harness and audit log, which run without operator attention — from the operator-facing control plane proper; the schematic combines them because from the principal’s seat they present as one surface. To the right, four agent teams — product and engineering (Chap. 3), marketing and growth (Chap. 4), sales and support (Chap. 5), and finance and compliance (Chap. 6, Chap. 7) — emit work to customers and revenue. The feedback loop closes through telemetry, evaluation and the audit log back to the operator console, which is what converts activity into evidence (Chap. 11, and the evaluation apparatus of Chap. 12). The gauge on the far left is the autonomy level L0–L5 defined in Sect. 1.6: it is a design setting, chosen per workflow, not a property of the model.

1.1 Why firms have the size they do

The theory of the firm begins with an embarrassment. If markets allocate resources efficiently through prices, why is so much economic activity organised inside hierarchies that suppress prices and substitute authority? Coase [43] answered that using the market is not free: there are costs of discovering relevant prices, of negotiating and concluding a separate contract for each exchange, and of enforcing what was

agreed. Where those costs exceed the cost of directing the same activity administratively, the activity is brought inside a firm. The boundary of the firm sits where the two marginal costs meet. This is a comparative-institutional argument, not a technological one: it predicts firm size from the relative costs of two governance modes, not from the state of the production function. The programme that follows from it is Williamson's [249], and the empirical literature it generated is unusually favourable to the theory — though Masten [146] cautions that its practical significance rests on the cost of *mistakes* in organisational choice, which is precisely the quantity a solo operator must estimate when deciding what to delegate to machines.

Williamson [250] sharpened the argument into an operational research programme by shifting the unit of analysis from the firm to the transaction and asking which governance structure — market, hybrid, or hierarchy — economises on the combined costs of contracting for a transaction with given attributes. The decisive attributes are asset specificity, the frequency of the exchange and the uncertainty surrounding it. A transaction requiring investments that have little value outside the relationship, repeated often, under conditions where contingencies cannot be enumerated in advance, is badly served by a spot contract and better served by hierarchy, which can adapt by fiat and settle disputes internally. This is why, historically, the work most reliably pulled inside firms has been work that is idiosyncratic, continuous and hard to specify in a contract — precisely the work that a solo operator is now attempting to delegate to machines.

Hierarchy solves adaptation problems and creates new ones. Once the person bearing the residual risk is not the person doing the work, the interests of the two diverge, and the difference must be managed at a cost: Jensen and Meckling [110] decompose it into monitoring expenditure by the principal, bonding expenditure by the agent, and the residual loss that persists even when both have been optimally incurred. This decomposition is the direct ancestor of the cost model in Sect. 1.7, and it transfers to machine agents with unsettling ease. A delegated agent has no incentives to align in the human sense, but it does have an objective specified by someone else, an information advantage over its principal about what it actually did, and a capacity to produce plausible reports of success. Kolt [118] argues in exactly these terms that the classical instruments — incentive design, monitoring and enforcement — were built for human agents operating at human speed, and that new technical and legal infrastructure is required when the agent acts in milliseconds and at scale. Monitoring cost and residual loss do not disappear when the agent is software; they become the dominant terms.

The fourth pillar is coordination itself, treated as an object of study rather than as overhead. Malone and Crowston [144] define coordination as the management of dependencies between activities and catalogue the mechanisms — shared resources, producer–consumer relationships, simultaneity constraints — each of which imposes its own communication and decision cost. The value of this framing for the present argument is that it is technology-independent and therefore measurable. A firm's coordination cost is a function of how many dependencies exist between activities and how expensive each dependency is to manage. Information technology attacks the second factor; a solo operator attacks the first, by eliminating the interpersonal dependencies entirely and replacing them with dependencies between processes that one person specifies. That substitution has a long prehistory, set out in Fig. 1.3.

From Mechanisation to the Autonomous Firm

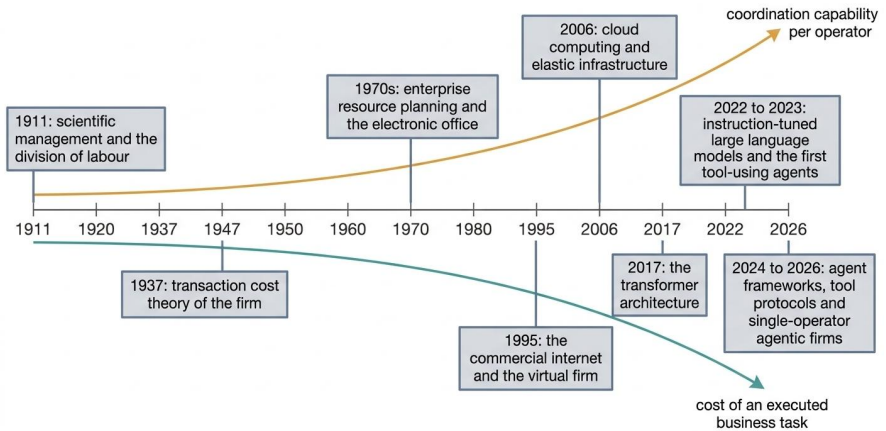


Figure 1.3: The historical arc that terminates in the single-operator agentic firm. Reading left to right: scientific management and the division of labour (1911) establish the decomposition of work into specifiable tasks; the transaction cost theory of the firm (1937) supplies the explanation for why those tasks are grouped inside organisations [43]; enterprise resource planning and the electronic office (1970s) instrument the internal coordination those groupings require; the commercial internet (1995) and elastic cloud infrastructure (2006) externalise capabilities that previously required owned assets and staff; the transformer architecture (2017) and instruction-tuned foundation models (2022–2023) make natural-language task specification executable [28]; and agent frameworks, tool protocols and control planes (2024–2026) make specification *operational* without an intermediating employee. The ascending curve above the axis is coordination capability per operator; the descending curve is the cost of one executed business task. The intersection of those two trends, not any single model release, is what makes the form of Fig. 1.2 feasible.

The sequence in the figure is instructive because each earlier step lowered coordination cost without shrinking firms. Enterprise systems, the commercial internet and cloud infrastructure all reduced the price of managing dependencies, and the empirical literature on transaction cost economics found that firms responded by doing more — expanding scope, outsourcing selectively, and reorganising internally — rather than by dissolving [37, 248]. The theory explains why. Firm size responds not to the level of coordination cost but to the *differential* between external and internal coordination cost. A technology that reduces both symmetrically leaves the boundary where it was. Contraction requires the external cost to fall faster than the internal one, so that market-mediated exchange displaces administrative direction at the margin.

What is different about agentic AI is that it attacks a term the previous waves left intact: the cost of *specifying and executing* non-routine work without a human intermediary. Agrawal et al. [5] model this as a decomposition of decisions into prediction and judgement, where machines absorb prediction and humans retain judgement; the economic consequence is that the value of human time concentrates in the judgement component, and the quantity of prediction one person can consume rises without bound. Acemoglu and Restrepo [3] supply the necessary counterweight: automation displaces labour from tasks, but the same technological frontier also creates new tasks, and the net effect on labour demand depends on which force dominates. For the solo firm the new tasks are internal and unavoidable — specification, evaluation, guardrail engineering, incident response — and Sect. 1.7 prices them.

Two recent contributions complete the theoretical picture in the direction this book takes. Wan [241] argues that generative AI opens a governance mode that is neither market nor hierarchy, because the coordination happens inside a shared representational space rather than through either prices or authority. Klein and Wieczorek [117] formalises the consequence as a domain-conditional unbundling: firm-size distributions shift toward micro-specialised execution capacity plus protocol-level orchestration, but only in domains where knowledge velocity is high, and only to the extent that verification cost can be held below the integration cost it replaced. Both predictions are conditional, and the condition is the subject of this book.

Proposition 1.1. The optimal size of a firm contracts under a coordination technology only if the technology reduces the cost of externally sourced or automatically executed work faster than it reduces the cost of internal administrative direction, and only if the residual verification and accountability costs it introduces are smaller than the coordination costs it removes. A solo enterprise is therefore viable in a domain not because agents are capable, but because verification in that domain is cheap relative to execution.

1.2 What actually changed: the delegation evidence

The theoretical case would be idle without evidence that delegation to language models moves real output. That evidence exists, is unusually good for so young a technology, and is more heterogeneous than the public discussion implies. It is worth stating the four principal results with their settings intact, because the settings are what determine whether a solo operator can rely on them.

The cleanest result concerns mid-level professional writing. Noy and Zhang [174] randomised college-educated professionals to perform occupational writing tasks with or without access to a general-purpose chatbot and found that average time on task fell by roughly 40% while output quality ratings rose by about 18%. Two features of the result matter more than the headline. First, the gain came predominantly from compressing time rather than from raising the ceiling of achievable quality — the tool substituted for effort on a first draft, not for expertise in the final judgement. Second, the effect was strongly skill-compressing: weaker writers improved far more than stronger ones, narrowing the distribution. For a solo firm this is exactly the profile that matters, because the operator will be a novice in most of the functions the firm has to perform.

The largest field study concerns customer support. Brynjolfsson et al. [32] observed the staged rollout of a conversational assistant to more than five thousand support agents and measured roughly a 15% average increase in issues resolved per hour, with the gains concentrated among the least experienced staff, who improved by around a third, while the most experienced agents gained little or nothing. The study also reports a small improvement in measured quality alongside the throughput gain. The mechanism the authors identify — the assistant propagates the implicit knowledge of high performers to everyone else — is precisely the mechanism a one-person firm depends on when it operates a support function without a support department.

The software engineering evidence is where external validity becomes acute. Peng et al. [196] randomised developers to complete a specified programming task (implementing an HTTP server in JavaScript) with or without an AI code assistant and measured a 55.8% reduction in completion time. That figure is the most cited productivity number in the field and the most context-dependent: the task was greenfield, self-contained, well-specified and of a kind heavily represented in training data. Controlled work on experienced open-source maintainers performing real tasks in large repositories they already know does not reproduce that magnitude [23], and the divergence is now the standard illustration of how strongly measured uplift depends on task novelty, codebase familiarity and the cost of verifying the result. A solo operator building a new product from scratch sits closer to the first setting; the same operator maintaining that product two years later sits closer to the second, and should plan capacity accordingly.

The most consequential result for the design of a solo firm is the consulting field experiment of Dell'Acqua et al. [49]. Consultants given access to a frontier model completed tasks 25.1% faster, produced 12.2% more of them, and scored roughly 40% higher on quality — when the tasks lay inside the model's capability boundary. On a task deliberately constructed to fall outside that boundary, participants using the model were substantially more likely to reach the wrong answer than those working without it. The authors name the boundary the **jagged technological frontier**: capability is not a smooth function of apparent task difficulty, so a human cannot infer from the ease of one delegated task that a superficially similar one will also succeed.

The jagged frontier: delegation that feels productive while degrading quality

The failure mode is not that the agent refuses or errs visibly. It is that output arrives faster, in the expected format, with confident supporting argument, and is wrong in a way the operator is not equipped to detect — because the operator delegated the task precisely in order not to have to acquire that expertise [49]. Three properties make this the characteristic hazard of the one-person firm. First, the frontier is *jagged*: adjacent tasks of similar apparent difficulty fall on opposite sides of it, so favourable experience is not evidence of general reliability. Second, the operator's subjective signal is inverted — speed and fluency rise as accuracy falls, so the felt experience of a bad delegation is indistinguishable from a good one. Third, there is no colleague. In a firm with staff, an out-of-frontier error passes several people who each hold a partial model of what correct looks like; in a solo firm it passes one person who has already decided not to hold that model. The operational response is not scepticism as a mood but verification as an engineered artefact: task-level acceptance tests, adversarial checks, and a standing rule that any workflow without an automatic correctness signal must be treated as one autonomy level lower than its nominal setting (Chap. 9, Chap. 12).

Diffusion evidence completes the picture. Bick et al. [27] surveyed the United States working-age population and found 39.4% had used generative AI by August 2024, with 28% of employed respondents using it for their job, 24.2% using it for work at least one day in the preceding week, and 10.6% using it every working day. Their comparison is the useful one: workplace adoption reached 28% within roughly two years of the technology becoming publicly available, against about 25% for the personal computer over three years at a comparable stage. Individual adoption is not organisational deployment, and the gap between the two is where the opportunity for a solo firm lies: the tools are universally available, while the operating architecture to run a business on them is not. The annual sector-wide review of the field reaches the same structural conclusion from aggregate data on capability, cost and deployment [220], and it is the reference to consult for the current-year version of every capability and price figure quoted in this book. A systematic synthesis of the productivity literature to date [216] finds the same qualitative pattern across studies — real, sizeable, task-level gains, concentrated in less experienced workers, with wide dispersion across settings and little evidence yet on firm-level aggregation.

Table 1.1 summarises the evidence base with the caveat attached to each result, because in this literature the caveat carries as much design information as the point estimate.

Three honest limitations should be stated before these numbers are used for planning. First, every effect size in Table 1.1 was measured on a single task type inside an organisation that supplied division of labour, quality assurance and escalation paths; nothing in the literature measures one person running engineering, marketing, support and finance concurrently. Second, the studies measure task-level productivity, and whether task-level gains survive aggregation into firm-level output — rather than dissipating in verification, rework and coordination — remains unre-

Table 1.1: Principal evidence on delegating knowledge work to language models. Effect sizes are as reported by the cited studies in their own settings; none of them was measured on a solo operator running multiple business functions simultaneously, which remains the central external-validity gap for the form described in this book.

Setting	Design	Reported effect	Caveat for a solo firm
Professional writing [174]	Randomised, college-educated professionals, occupational writing tasks	Time on task -40% ; quality ratings $+18\%$; distribution compressed	Gains are in draft speed, not in final judgement; the operator still owns the acceptance decision
Customer support [32]	Field study, staged rollout, $>5,000$ agents	$\approx +15\%$ issues resolved per hour; novices up to $\approx +35\%$; small quality gain	Gains accrue to novices; an expert operator should not expect the headline number on their own strongest function
Software development [196]	Randomised controlled trial, specified greenfield task	Completion time -55.8%	Greenfield, well-specified, heavily represented in training data; a ceiling estimate, not a planning figure
Experienced maintainers [23]	Controlled study, real tasks in developers' own large repositories	Does not reproduce the greenfield speed-up	Maintenance of an existing product is the solo firm's steady state, so this is the more relevant regime
Consulting tasks [49]	Field experiment with in-frontier and out-of-frontier tasks	-25.1% time, $+12.2\%$ tasks, $\approx +40\%$ quality in frontier; accuracy degraded outside it	The frontier is jagged and unobservable from inside; verification must be designed, not improvised
Adoption [27]	Population survey, United States, ages 18–64	39.4% ever used; 28% used for work; 10.6% daily at work	Availability of tools is not an operating architecture; diffusion of use precedes diffusion of capability

solved [117, 216]. Third, the results are technology-dated: they were obtained with particular model generations under particular interfaces, and capability, price and failure modes have all moved since. The correct use of this evidence is as a set of order-of-magnitude priors for capacity planning, tested against the operator's own instrumented measurements (Chap. 11), not as a warranty.

1.3 The one-person economy in numbers

The one-person firm is not an exotic organisational experiment; it is the modal business form in the largest economy in the world. The United States Census Bureau's Nonemployer Statistics programme, an administrative-data census of businesses with no paid employees and at least \$1,000 in annual receipts, counted 30.4 million such businesses generating close to \$1.8 trillion in receipts in 2023, up from 29.8 million the year before [232]. On the Bureau's own accounting, businesses with no paid employees account for 78.4% of all United States businesses. The peer-reviewed treatment of the series confirms that this is structural growth rather than a statistical artefact of definitional change or of the business cycle [44].

The arithmetic implied by those two aggregates is the first quantitative fact this book must confront. Dividing \$1.8 trillion of receipts by 30.4 million establishments gives average annual receipts of approximately \$59,000 per nonemployer business. The modal one-person firm, in other words, is not a leveraged software company; it is a sole trader with modest turnover, and a substantial share of the population earns considerably less than the mean. Any claim that AI transforms solo enterprise has to be measured against that baseline rather than against the outliers, and the distance between the two is the subject of Sect. 1.9.

Independent work is also growing on the paid-platform side, where the composition of demand has shifted toward AI-related skills. Upwork's platform research reports that freelancers performing AI-related work earn materially more per hour than those who do not, and that AI-related work on the platform has grown rapidly to exceed several hundred million dollars annualised [230]. Two cautions apply. Platform data describe one marketplace, not the labour market. And the attribution problem is severe: the growth of solo self-employment in AI-exposed occupations cannot presently be separated from tax-driven entity formation, gig-platform effects and post-pandemic labour-market composition. I therefore treat the direction of travel as established and decline to attach a causal coefficient to it.

On the adoption side, the official statistics are sobering and should be read against vendor surveys with scepticism. The Census Bureau's Business Trends and Outlook Survey, a biweekly instrument covering approximately 1.2 million businesses, put overall business use of AI for any business function in the high teens as of late 2025 and early 2026, and use of AI to produce the goods or services the business actually sells at only a few per cent [233]. Adoption rises with firm size: the Bureau's reporting identifies larger firms as the heaviest users, and its dedicated small-business research establishes the official baseline for the smallest firms [231]. The Federal Reserve maintains a parallel macro-statistical series for the same purpose [8]. The gap between "uses AI for some function" and "uses AI to produce what it sells" is the single most important measurement caveat in this literature, and it is also the gap this book is about closing: the one-person AI company is defined by production,

not by assistance. Against that baseline, Fig. 1.4 states the leverage the form has to deliver to be worth its overhead.

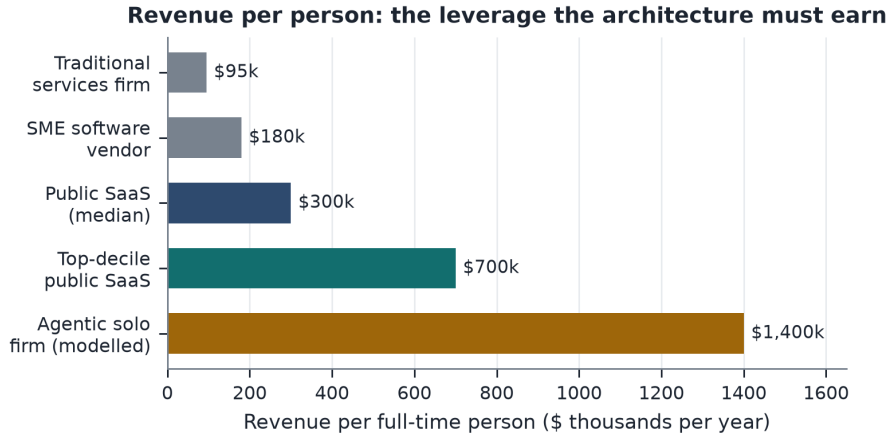


Figure 1.4: The benchmark frame for revenue per person that the architecture of this book must earn. The first four bars — a traditional services firm at \$95k, a small or mid-sized software vendor at \$180k, a median public software company at \$300k, and a top-decile public software company at \$700k per full-time person per year — are *order-of-magnitude reference points drawn from published benchmark ranges*, rounded for legibility, and are not observed averages of any specific sample; published benchmarks for private software companies cluster near \$141k median with an average near \$300k for listed companies, while AI-native outliers are reported an order of magnitude higher again [25, 194, 208]. The fifth bar, \$1,400k, is the *modelled target of this book*: the revenue per person that a single-operator agentic firm must reach for the governance overhead of Sect. 1.7 to be worth incurring. It is a design target, not an observed average, and no claim is made that a typical solo firm attains it — the mean nonemployer business earns about \$59k of receipts per year [232].

The target in the figure deserves to be read defensively. A practitioner benchmark of private software companies puts median revenue per employee at roughly \$141,000 [208]; the figure commonly quoted for the average listed software company, near \$300,000, comes from trade coverage rather than from audited aggregates [194], and the two are not measured on the same basis. Third-party estimates of frontier AI-native firms report figures an order of magnitude above that again, and the most-cited small-team cases — an image product generating substantial revenue with a team reported in the low tens, a one-person photographic product reported at six-figure monthly recurring revenue — are widely circulated [107, 194, 210]. Every one of those figures is a third-party or self-reported estimate rather than an audited disclosure, headcount denominators differ in whether contractors and agencies are counted, and the sample is selected on success. Chap. 11 treats the case evidence properly, including the base rates that publicised cases conceal. For the purposes of this chapter the benchmark frame has one job: to fix the order of magnitude of leverage that a solo architecture must deliver, so that later chapters can be evaluated

against a number rather than against an aspiration.

1.4 Defining the one-person AI company

Definitions in a young field are load-bearing engineering artefacts rather than scholarly courtesies. The term used throughout this book denotes a governance form, not a technology stack and not a revenue level.

Definition 1.2 (One-person AI company). A **one-person AI company** is a commercial enterprise in which (i) exactly one natural person holds strategic authority and legal accountability for the firm's obligations; (ii) the substantial majority of recurring execution across the firm's business functions is performed by automated systems — models, workflows and agents — rather than by human labour, whether that labour is employed or contracted; and (iii) the human principal exercises control through a designed control plane consisting of objectives, policies, budgets, approval gates, evaluation and an audit trail, rather than through direct production of the firm's output.

Each clause excludes something. The first excludes partnerships and founder-plus-contractor arrangements, not because they are inferior but because their control problem is different: they have a second locus of judgement. The second excludes firms in which automation assists a human who still performs the work; on the Census distinction of Sect. 1.3, the form requires AI in the production of what the business sells, not merely in some business function [233]. The third is the substantive clause and the hardest to satisfy. A firm whose control consists of the operator reading outputs and reacting to complaints has no control plane; it has a habit. The presence of an explicit, inspectable control surface — what the agent may do, with which resources, up to what expenditure, subject to which review, evidenced how — is what distinguishes the form from improvisation, and it is the reason the architecture in Fig. 1.2 places the control plane at the top of the stack rather than at the side.

The form has three structural components. The **human principal** supplies objectives, judgement on irreversible decisions, and the signature on contracts, filings and representations. The **delegated execution layer** is the set of models, tools, memories, workflows and agents that convert objectives into delivered work; it is heterogeneous by design, mixing deterministic code where determinism is available with model-driven decision where it is not. The **control plane** is the mechanism through which the principal's authority is expressed as machine-enforceable constraint and through which the execution layer's behaviour is made legible: policy, budget, gating, evaluation, telemetry and audit log. The remainder of this book is substantially an engineering treatment of the second and third components, because the first is a person and cannot be scaled.

Principle 1.3 (The accountability invariant). Authority may be delegated; accountability may not. Every action of the delegated execution layer is, in law and in commercial practice, an action of the human principal. Delegation therefore transforms the principal's obligations — from performing the work to specifying, constraining, verifying and evidencing it — but it never transfers them. An architecture that cannot reconstruct, after the fact, what was done on the firm's behalf, under whose authority,

within which limits, has not reduced the principal's liability; it has only reduced the principal's knowledge of it.

Principle 1.3 is not a moral exhortation. It is the design constraint from which most of the technical apparatus of this book follows, and it has direct support in both the legal-academic and the regulatory literature. The principal-agent analysis of AI delegation concludes that the information asymmetry between a fast-acting agent and its principal must be closed by technical infrastructure — visibility, logging, attributability — because the classical human instruments do not transfer [118]. The liability literature reaches the same place from the statutory side: as autonomy increases, the allocation of responsibility across the user, integrator and developer chain shifts, and the proposed instruments are mandatory logging of agent decisions, clear indication of agent-driven action, and duty-of-care standards indexed to autonomy [132]. In a solo firm every one of those roles except model development collapses onto one person, so the practical consequence of Principle 1.3 is that the audit log is not observability infrastructure; it is the firm's evidentiary record. Chap. 7 develops the contractual and intellectual-property consequences, and Chap. 8 the integrity requirements that make a log worth presenting.

Two negative definitions prevent the most common misreadings. The form is *not* a solo consultant with a chatbot. A consultant who drafts faster is a consultant with better tools: the throughput ceiling is still the consultant's own hours, and the productivity gain is bounded by the effect sizes of Table 1.1 applied to personally performed work — worth having, but arithmetically incapable of producing the leverage in Fig. 1.4. The form requires that work proceed while the principal is not present, which is a statement about architecture, not about enthusiasm. Equally, the form is *not* an unsupervised autonomous agent operating a business. Full autonomy in the sense of goal-setting and self-modification is explicitly not recommended for enterprise deployment by the practitioner community that has thought hardest about the control problem [112], and the empirical failure record — agent compromise, agent injection, memory and context poisoning, cascading multi-agent failure — makes the reason concrete rather than philosophical [156, 182]. The one-person AI company sits between those poles by construction: maximal delegation of execution, retained and instrumented human authority.

Remark 1.4. The form is defined by the locus of accountability, not by a headcount of zero. A one-person AI company may purchase professional services — an accountant, a lawyer, a penetration test — without ceasing to be one, because those suppliers do not hold the firm's execution loop. What it may not do and remain within Definition 1.2 is maintain a second person who exercises independent operational judgement over the firm's recurring work.

1.5 Workflows and agents, and why the distinction is load-bearing

The most consequential architectural decision in this book is also the least glamorous: for a given piece of recurring work, does the sequence of steps get decided in advance by code, or at run time by a model? The canonical engineering statement of the distinction is control-flow based, and this book adopts it without modification [59].

Definition 1.5 (Workflow). A **workflow** is a system in which models and tools are orchestrated through predefined code paths. The set of possible execution paths is enumerable before run time; the model supplies content, classification or judgement at fixed points, and the sequencing, error handling and termination conditions are properties of the surrounding program.

Definition 1.6 (Agent). An **agent** is a system in which a model dynamically directs its own process and tool use, retaining control over how a task is accomplished. The execution path is chosen at run time as a function of intermediate results, so the set of realisable paths is not enumerable in advance and grows with the horizon of the task.

The distinction is load-bearing for three reasons, none of which is about elegance. The first is testability. If the path set is enumerable, the system can be tested by enumerating paths and asserting behaviour on each; if it is not, testing becomes sampling from a distribution over behaviours, and assurance becomes statistical. NVIDIA's security taxonomy makes the consequence explicit: at full autonomy the number of potential execution paths grows exponentially with the number of execution steps, so exhaustive path enumeration is impractical and the controls must change in kind — taint tracing and mandatory approval of sensitive actions rather than dataflow review [203]. The second reason is cost variance. A workflow consumes a predictable number of model calls; an agent's consumption is a random variable with a heavy tail, which is why Chap. 6 treats budget enforcement as a control-plane function rather than as a monthly reconciliation. The third is attribution. When a workflow produces a wrong result, the failing step is identifiable by construction; when an agent does, the operator must reconstruct a trajectory, which is only possible if the trajectory was recorded — again Principle 1.3.

The engineering literature converges on a small catalogue of composable workflow patterns that solve most recurring problems without a fully agentic loop: prompt chaining, routing, parallelisation, orchestrator-workers, and evaluator-optimiser, with autonomous agents treated as a distinct sixth category to be used when the others do not fit [59]. The academic surveys describe the same design space from the research side, cataloguing agent construction into profiling, memory, planning and action components [242], distinguishing single-agent task automation from multi-agent coordinated systems with persistent memory and dynamic task decomposition [212], and observing across ninety studies that the choice between deterministic, plan-based execution and generative, stochastic execution tracks the criticality of the domain: safety-critical domains stay symbolic, data-rich commercial domains go generative [2]. The reasoning-and-acting loop that underpins most practical agents [257] and the multi-agent conversation frameworks built on it [198, 252] are treated as implementation substrate in Chap. 2.

For a one-person firm the operational rule that follows is a preference, not a prohibition. Use a workflow wherever the path set is known, because a workflow converts a supervision problem into a testing problem, and testing is capital that compounds while supervision is expense that recurs. Reserve agents for work whose decomposition genuinely depends on what is discovered along the way — diagnosis, research, negotiation of ambiguity — and pay for them with explicit containment: bounded tool scope, bounded budget, bounded time, mandatory approval on irreversible ef-

fects. A large fraction of what is marketed as agentic in commercial practice is a workflow with a language model in one of its steps, and that is usually the correct engineering choice. The distinction earns its place in this chapter because it determines which autonomy level a given piece of work can safely be assigned, which is the subject of the next section.

1.6 Levels of enterprise autonomy

Autonomy is a design decision, separate from capability and from operating environment [75]. This is the single most important conceptual move in the taxonomy literature and it deserves emphasis: a highly capable model may be deployed at low autonomy, and a weak one at high autonomy, and the second combination is the dangerous one. Consequently the ladder below classifies *deployments*, per workflow, and a single firm will operate at several levels simultaneously — L4 for content drafting, L1 for anything that touches a customer contract.

Fig. 1.5 shows the ladder. Each level is defined by four attributes: who holds the control loop, what the human does per unit of work, the blast radius of an uncorrected error, and the governance obligation that the level creates.

At **L0 (Manual)** the human holds the control loop completely and software is a passive instrument: spreadsheets, editors, search. Every unit of work consumes human time proportionally, blast radius is confined to what one person can do in an afternoon, and the governance obligation is ordinary professional diligence. At **L1 (Assisted)** the model generates candidate output and the human edits every item before it has any external effect. The control loop is still human; the productivity gain is precisely the effect measured in the writing and support studies of Sect. 1.2 [32, 174]. Blast radius remains small because nothing reaches the world unread, and the governance obligation is limited to disclosure where disclosure is required and to record-keeping of sources.

At **L2 (Delegated Tasks)** an agent or workflow completes a bounded task end to end — draft a reply, produce a report, open a pull request — and the human approves each completed task before it takes effect. The control loop is shared: the machine holds it within the task, the human at the boundary. This is the level at which review cost begins to dominate, because approval time scales with the number of tasks while the operator’s hours do not. Blast radius is one task’s worth of error, and the governance obligation adds an explicit record of what was approved by whom. At **L3 (Supervised Workflows)** the machine holds the control loop for a whole workflow and the human sees only exceptions: items the workflow flags, items that violate policy, items whose confidence or cost falls outside an envelope. This is the first level at which the operator’s time stops scaling with volume, and it is where most of the economic value of the form is realised (Sect. 1.7). It is also the first level requiring real control engineering: exception criteria, escalation paths, budget caps, and an audit trail sufficient to reconstruct unreviewed work.

At **L4 (Managed Functions)** agent teams own an entire business function within declared budget and policy limits, and the human sets objectives and reads aggregate results. The control loop is machine-held across many workflows; the human intervenes on trend, not on instance. Blast radius is a function’s worth of damage — a mispriced campaign, a month of degraded support, a compliance omission repeated

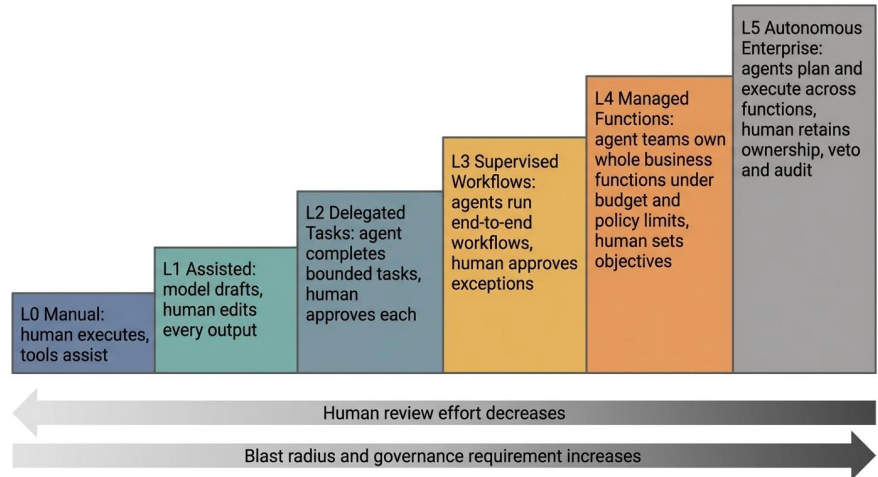


Figure 1.5: The L0–L5 enterprise autonomy ladder used throughout this book. L0 Manual: the human executes and tools assist. L1 Assisted: the model drafts and the human edits every output. L2 Delegated Tasks: an agent completes bounded tasks and the human approves each one. L3 Supervised Workflows: agents run workflows end to end and the human approves exceptions only. L4 Managed Functions: agent teams own whole business functions within budget and policy limits while the human sets objectives. L5 Autonomous Enterprise: agents plan and execute across functions, with the human retaining ownership, veto and audit. The two opposing gradients along the base are the crux of the design problem: human review effort per unit of work falls as the level rises, while blast radius and the governance obligation rise. Levels apply to a deployment, not to a firm and not to a model; the autonomy level of a workflow is chosen, and Sect. 1.7 shows the choice has an interior optimum.

across every transaction — and the governance obligation escalates correspondingly: pre-authorised action classes, enforced expenditure ceilings, continuous evaluation, and periodic assurance that the enforcement itself still works. At **L5 (Autonomous Enterprise)** agents plan and execute across functions, including the allocation of resources between them, while the human retains ownership, veto and audit. It is important to be candid that L5 is an analytical endpoint rather than a recommendation: the practitioner scale on which this ladder is partly modelled explicitly advises against enterprise deployment at full autonomy [112], and the academic scheme's highest level is defined as operation that “does not require, and comes with no means for” user involvement [75] — a property that is incompatible with Principle 1.3 when the operator remains liable.

1.6.1 Why the published scales do not align

A reader coming to this ladder from the standards and vendor literature will immediately want a mapping to the scales already in circulation. That mapping can be sketched, but it must be sketched honestly, because the scales anchor on different variables. The Cloud Security Alliance's six-level enterprise scale anchors on the approval relationship, from no autonomy through assisted, supervised, conditional and high autonomy to full autonomy including self-modification, and is explicitly modelled on the automotive levels-of-driving-automation taxonomy [112]. NVIDIA's four-level scale anchors on control-flow structure and tool access — single inference call, deterministic multi-step sequence, weakly autonomous tool selection at fixed decision points, fully autonomous tool use with plan revision — and attaches an escalating control set to each level [203]. The five-level academic scheme anchors on the human's role, naming levels Operator, Collaborator, Consultant, Approver and Observer, and proposes third-party autonomy certificates supported by evidence-based autonomy cases as the governance instrument [75]. The European liability classification anchors on three dimensions at once — generality of function, control mechanism and tool access — and maps levels to the allocation of liability across users, integrators and developers [132]. Statute, meanwhile, declines to number anything: the EU AI Act defines an AI system as one designed to operate with “varying levels of autonomy” and specifies no levels at all [68].

Table 1.2 sets the four scales beside this book's ladder. The discrepancies are informative rather than merely inconvenient. NVIDIA's scale saturates at three because control-flow freedom saturates: once a model may freely choose whether, when and how to invoke tools and may revise its plan, there is no further structural freedom to grant, even though a firm can still delegate more *scope*. This book's L4 and L5 both map to that single NVIDIA level, which tells the operator something useful: the security controls required at L4 and at L5 are the same controls, and the difference between the levels is economic and organisational, not technical. Conversely the academic five-role scale distinguishes Consultant from Approver in a way that maps awkwardly onto this ladder's L3, because a solo operator is simultaneously the consultant's client and the approver of record. The absence of a legal mapping is the most consequential gap of all: no numbered autonomy level currently triggers a defined statutory obligation, so the operator cannot discharge a duty by declaring

Table 1.2: Approximate crosswalk between this book’s L0–L5 enterprise autonomy ladder and the principal published scales. **The scales do not align and there is no accepted crosswalk.** They anchor on different variables: approval relationship (Cloud Security Alliance, six levels, modelled on the automotive driving-automation taxonomy), control-flow structure and tool access (NVIDIA, four levels), the human’s role (Feng et al., five levels), and liability allocation (Interface/Ada Lovelace, five levels by three dimensions). The EU AI Act supplies no numbered levels at all, defining an AI system as one operating with “varying levels of autonomy” [68], so no legal mapping from a level to a concrete obligation exists. This table is an orientation aid, not an equivalence claim; do not use it to transfer a control requirement from one scale to another without re-deriving it.

This book	CSA 0–5 [112]	NVIDIA 0–3 [203]	Five roles [75]	Liability tilt [132]
L0 Manual	L0 No autonomy	L0 Inference API	(below scale)	User
L1 Assisted	L1 Assisted	L0–L1 Deterministic	L1 Operator	User
L2 Delegated tasks	L2 Supervised	L1–L2 Weakly autonomous	L2 Collaborator	User, integrator emerging
L3 Supervised workflows	L3 Conditional	L2 Weakly autonomous	L3 Consultant	Shifts to integrator
L4 Managed functions	L4 High autonomy	L3 Fully autonomous	L4 Approver	Integrator and developer
L5 Autonomous enterprise	L5 Full autonomy (not recommended)	L3 Fully autonomous	L5 Observer	Developer and provider

a level, and must instead document what the level means in their own deployment (Chap. 7).

An autonomy level is only real if it is enforced by something other than intention. In this book the level of every workflow is declared in machine-readable policy that the control plane enforces at the point of action; Listing 1.1 shows the shape of such a declaration, and Chap. 9 develops the full gating apparatus.

```

1 version: 1
2 firm: solo-operator
3 defaults:
4   fail: closed # unmatched action is denied, not allowed
5   audit: required # no action without an audit record
6   attribution: principal # every action is an act of the principal
7
8 workflows:
9   - name: content-production
10     autonomy: L4 # agent team owns the function
11     scope:
12       tools: [research.search, cms.draft, cms.publish, image.generate]
13       data: [public, marketing-owned]
14     limits:
15       spend_month_usd: 400
16       actions_per_hour: 60
17       irreversible: none # publishing is revocable within 24h
18     review:
19       sample_rate: 0.05 # 5% of published items reviewed post hoc
20       escalate_when: [policy.brand_violation, eval.score < 0.72]
21       evidence: [prompt, sources, model_version, diff, approver]
22
23   - name: customer-contracting
24     autonomy: L1 # model drafts, human edits every output
25     scope:
26       tools: [crm.read, docs.draft]
27       data: [customer-pii, contracts]
28     limits:
29       spend_month_usd: 40
30       actions_per_hour: 10
31       irreversible: all # signature and pricing are irreversible
32     review:
33       sample_rate: 1.00 # every item reviewed before effect
34       escalate_when: [always]
35       evidence: [prompt, draft, human_edit_diff, signature_record]

```

Listing 1.1: Declared autonomy level, blast-radius limits and approval gates for two workflows in a one-person firm. The level is a property of the deployment, not of the model: the same model serves an L4 content workflow and an L1 contract workflow. Enforcement is at the point of action, so an undeclared tool or an over-budget step fails closed.

Policy declared is not policy enforced. Listing 1.2 shows the authorisation function that stands between an agent’s intent and an effect on the world. It is deliberately unremarkable code: the intellectual content of a control plane is in the completeness of the checks and the immutability of the record, not in cleverness.

```

1 from dataclasses import dataclass

```

```

1  # Highest autonomy level permitted to act unattended, by
2  # reversibility class of the requested action.
3  UNATTENDED_CEILING = {
4      "reversible": 3, # L3+ may act without per-item review
5      "costly": 4, # L4+ only, and inside spend envelope
6      "irreversible": 6, # never unattended: 6 is unreachable
7  }
8
9
10
11
12 @dataclass(frozen=True)
13 class Action:
14     workflow: str
15     tool: str
16     reversibility: str # reversible | costly | irreversible
17     est_cost_usd: float
18
19
20 class Denied(Exception):
21     pass
22
23
24 class Gate:
25     def __init__(self, policy, ledger, audit):
26         self.policy = policy # parsed autonomy declaration
27         self.ledger = ledger # month-to-date spend per workflow
28         self.audit = audit # append-only, hash-chained
29
30     def authorise(self, act: Action) -> str:
31         wf = self.policy[act.workflow]
32         level = int(wf["autonomy"][1:])
33         decision, reason = "allow", "within declared envelope"
34
35         if act.tool not in wf["scope"]["tools"]:
36             decision, reason = "deny", "tool outside declared scope"
37         elif self._would_breach_budget(wf, act):
38             decision, reason = "deny", "monthly spend envelope"
39         elif level < UNATTENDED_CEILING[act.reversibility]:
40             decision, reason = "escalate", "level below ceiling"
41
42         self.audit.append(action=act, decision=decision,
43                           reason=reason, level=level)
44         if decision == "deny":
45             raise Denied(f"{act.tool}: {reason}")
46         return decision # "allow" or "escalate"
47
48     def _would_breach_budget(self, wf, act) -> bool:
49         spent = self.ledger.month_to_date(act.workflow)
50         return spent + act.est_cost_usd > wf["limits"]["spend_month_usd"]

```

Listing 1.2: The authorisation gate that converts a declared autonomy level into an enforced one. Actions are denied unless the tool is in declared scope, the spend envelope holds, and the level permits unattended execution of the action’s reversibility class. Every decision — allow, deny or escalate — is written to the audit log before the effect occurs, which is what makes Principle 1.3 operable.

Two properties of Listing 1.2 deserve comment because they recur throughout

the book. The gate is a *choke point*: authority is expressed in exactly one place, so raising or lowering an autonomy level is a configuration change rather than a code change scattered across workflows. And the audit record is written before the effect, not after, so an action that crashes mid-execution still leaves evidence that it was attempted. Both properties exist to serve Principle 1.3 rather than to serve reliability, which is why they belong in a chapter about paradigm rather than in an appendix about implementation.

Two people at \$480,000, or one person at L3

Consider two firms with identical annual revenue of \$480,000 in advisory work. All figures below are modelling assumptions used to expose structure, not observed accounts.

Traditional two-person consultancy. A principal and an analyst, each with 1,800 working hours per year (45 weeks at 40 hours), give 3,600 person-hours of capacity and revenue per person-hour of $480,000 / 3,600 = \$133.33$. The analyst costs \$95,000 fully loaded; premises, software, insurance and professional fees cost \$30,000. The residual to the principal is $480,000 - 95,000 - 30,000 = \$355,000$. Coordination is interpersonal: the principal spends about five hours a week briefing and reviewing the analyst, which is 225 hours a year, or 12.5% of the principal's capacity.

L3 solo operation. One operator with 1,800 hours gives revenue per operator hour of $480,000 / 1,800 = \$266.67$, twice the two-person figure. Delegated execution costs \$25,200 of inference (\$2,100 a month), \$6,000 of evaluation and observability, and \$12,000 of assurance — legal review of automated outputs, professional indemnity reflecting automated delivery, and audit-log tooling — for \$43,200 total, or 9.0% of revenue. Other overheads fall to \$18,000. The residual to the principal is $480,000 - 43,200 - 18,000 = \$418,800$, an improvement of \$63,800, or 18.0% on the two-person residual.

What the comparison actually shows. The supervision burden does not disappear. At L3 the operator reviews exceptions, maintains evaluation suites and investigates incidents; on the same five-hours-a-week assumption that is the same 225 hours, redirected from supervising a colleague to supervising a system. The differences are that the redirected hours produce a reusable artefact — tests, policies, guardrails — rather than a transient briefing, that the marginal cost of the next unit of delivery is a few dollars of tokens rather than a fraction of a salary, and that the firm's single point of failure is now unambiguous. The two-person firm can survive its principal taking a month off. The solo firm survives only if the control plane does.

1.7 The delegation calculus and the interior optimum

The autonomy ladder invites a naive strategy: climb it. That strategy is wrong, and the reason is arithmetic rather than temperament. Raising autonomy substitutes one cost for two others. Human labour cost falls, because fewer operator minutes are consumed per unit of delivered work. Governance and assurance cost rises, because

higher autonomy requires guardrail engineering, evaluation suites, monitoring, taint tracing on tool use, incident response capability and periodic assurance that the controls still function [203]. Residual expected loss also rises, because more work reaches the world without a human having examined it, and the expected value of uncorrected error is the product of its probability and its severity — which is, not coincidentally, the definition of risk used by the governing regulation [68]. The sum of the three has a shape, and the shape has an interior minimum.

Model 1.7 (Cost of operating at autonomy level a). For a fixed volume of delivered work, the total cost of operating at autonomy level $a \in \{L0, \dots, L5\}$ is

$$C(a) = C_L(a) + C_G(a) + \mathbb{E}[L](a), \quad (1.1)$$

where $C_L(a)$ is human labour cost — the operator hours consumed in producing, reviewing and approving the work; $C_G(a)$ is governance and assurance cost — control engineering, evaluation, monitoring, audit and incident capability; and $\mathbb{E}[L](a)$ is residual expected loss, the probability-weighted impact L of adverse events that the controls at level a fail to prevent. All three are stated as an index with the labour cost of operating entirely manually normalised to $C_L(L0) = 100$.

Eq. 1.1 is a decomposition, not a law: it asserts only that the three cost categories are exhaustive and that they move in opposing directions with a . Its ancestry is the agency-cost decomposition into monitoring expenditure, bonding expenditure and residual loss [110], with bonding expenditure — which a machine agent cannot incur on its own behalf — transferred to the principal as governance cost. The parameter values used here and plotted in Fig. 1.6 are calibrated judgement, not measurement, and they are stated explicitly so that a reader can substitute their own. Labour cost falls steeply and then flattens, because the first increments of delegation remove routine production while the last increments remove only oversight, which is harder to remove. Governance cost rises super-linearly, because each additional level requires the controls of the previous one plus a new class of control. Residual expected loss roughly doubles per level, reflecting both the growth in unreviewed volume and the widening blast radius of Fig. 1.5.

Table 1.3 gives the values, and Worked Calculation 1.1 works through the marginal reasoning that identifies the optimum. The qualitative result is robust to a good deal of reparameterisation and is the central practical claim of this chapter: for a firm that must bear the consequences of its automated actions, the cost-minimising autonomy level is interior. Pushing past it does not buy efficiency; it buys expected loss and control overhead at the price of labour savings that have already been mostly harvested.

Worked Calculation 1.1 Locating the cost-minimising autonomy level

Inputs (index; L0 labour = 100), from Table 1.3: $C_L = (100, 74, 52, 33, 18, 9)$; $C_G = (2, 6, 12, 22, 38, 62)$; $\mathbb{E}[L] = (1, 2, 4, 8, 17, 34)$ for L0...L5.

Step 1 — totals. Applying Eq. 1.1 level by level: $100 + 2 + 1 = 103$; $74 + 6 + 2 = 82$; $52 + 12 + 4 = 68$; $33 + 22 + 8 = 63$; $18 + 38 + 17 = 73$; $9 + 62 + 34 = 105$. So $C = (103, 82, 68, 63, 73, 105)$.

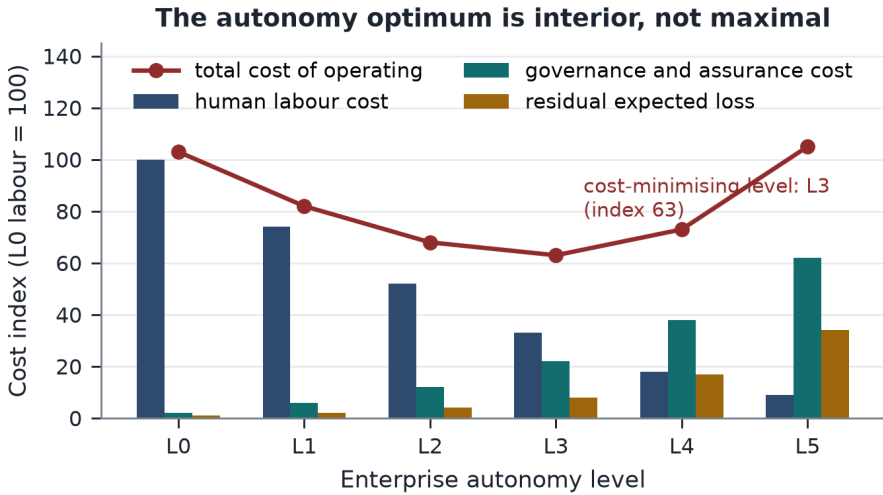


Figure 1.6: The autonomy optimum is interior, not maximal. Bars show the three components of Eq. 1.1 at each level, with the labour cost of manual operation normalised to 100: human labour $C_L = (100, 74, 52, 33, 18, 9)$ for L0 through L5; governance and assurance $C_G = (2, 6, 12, 22, 38, 62)$; residual expected loss $E[L] = (1, 2, 4, 8, 17, 34)$. The line is the total $C(a) = C_L + C_G + E[L] = (103, 82, 68, 63, 73, 105)$, which attains its minimum of 63 at L3 — supervised workflows with exception-only human review. The total at L5 (105) slightly exceeds the fully manual total (103): under these parameters, maximal autonomy is more expensive than no autonomy at all. The optimum’s location is a function of the parameters, not a universal result; Sect. 1.7 shows how improvements in verification technology move it rightward.

Table 1.3: Cost index by autonomy level under Model 1.7, with the labour cost of manual operation normalised to 100. These are the exact values plotted in Fig. 1.6. The successive marginal costs $C(a) - C(a - 1)$ are $-21, -14, -5, +10$ and $+32$: the first three increments of autonomy pay for themselves, the fourth does not, and the fifth is heavily loss-making. The minimum lies at L3.

Level	Labour C_L	Governance C_G	Expected loss $E[L]$	Total $C(a)$
L0 Manual	100	2	1	103
L1 Assisted	74	6	2	82
L2 Delegated tasks	52	12	4	68
L3 Supervised workflows	33	22	8	63
L4 Managed functions	18	38	17	73
L5 Autonomous enterprise	9	62	34	105

Step 2 — marginal test. The increments are $\Delta C = (-21, -14, -5, +10, +32)$. Autonomy should be increased while $\Delta C < 0$. The last profitable step is L2 $\rightarrow$ L3, worth -5 ; the next step is $+10$ and must be refused. The minimum is $C(L3) = 63$, a 38.8% reduction on manual operation $((103 - 63)/103 = 0.388)$.

Step 3 — why the L3 $\rightarrow$ L4 step fails. It saves $33 - 18 = 15$ of labour but adds $(38 - 22) + (17 - 8) = 16 + 9 = 25$ of governance and expected loss. The saving is real; it is simply smaller than what it costs to make the saving safe.

Step 4 — the endpoint. $C(L5) = 105 > C(L0) = 103$. Under these parameters full autonomy is more expensive than doing everything by hand, because the last two levels convert almost all remaining labour savings into control apparatus and residual risk.

Sensitivity. Doubling every residual loss term to (2, 4, 8, 16, 34, 68) gives totals (104, 84, 72, 71, 90, 139): the optimum stays at L3 but its advantage over L2 narrows to 1 index point. Halving governance cost to (1, 3, 6, 11, 19, 31) gives (102, 79, 62, 52, 54, 74): still L3, now nearly tied with L4 at 52 against 54. The decisive lever is verification. If better evaluation and containment cut the L4 governance term from 38 to 30 and its expected loss from 17 to 6, then $C(L4) = 18 + 30 + 6 = 54 < 63$ and the optimum moves to L4. This is the whole investment thesis of Chap. 9 and Chap. 12: money spent making verification cheaper and containment tighter does not merely reduce cost at a fixed level, it relocates the frontier.

Choosing a level for one workflow

Assign a level per workflow using four questions, in this order. Is there an automatic correctness signal — a test, a schema, a reconciliation, a customer confirmation — that fires without the operator looking? If not, the workflow cannot exceed L2, whatever the model can do. Is the worst realisable single action reversible within a period the operator will certainly notice? If not, the irreversible actions must be carved out to L1 even if the rest of the workflow runs at L4, exactly as in Listing 1.1. Does the expected volume make per-item approval infeasible on the operator's hours? If yes, L3 is mandatory rather than optional, and the exception criteria become the real product. Finally, does the level's governance obligation actually exist as built artefacts today — not as an intention? If not, the workflow runs one level lower until the artefacts exist. Levels are claims about installed controls, and a claim without controls is the failure mode described in Chap. 8.

1.8 The three scarce resources: verification, authority, attention

If the classical firm's scarce resources were capital and labour, the solo agentic firm's are verification, authority and attention. This is the central thesis of the book, and every subsequent chapter can be read as an attempt to relax one of the three constraints.

Verification is the capacity to establish that delegated work is correct before it has consequences, at a cost per item far below the cost of doing the work. It is scarce because correctness is domain-specific and cannot be generically purchased: a test suite verifies code, a reconciliation verifies a ledger, a citation check verifies a claim, and nothing verifies a strategic judgement. The jagged frontier makes verification the binding constraint rather than capability [49]: an operator who cannot verify an output cannot safely accept it, no matter how capable the system that produced it, and the subjective confidence that accompanies fluent output is not evidence. Verification is also the term through which the whole cost model runs. In Eq. 1.1 it appears twice, as the labour cost of reviewing and as the governance cost of automating the review, and Worked Calculation 1.1 showed that improving it is the only intervention that moves the optimum. The formal treatment of firm boundaries under agentic AI reaches the same conclusion from the theory side: verification cost scales with task throughput, which means it, not integration, becomes the growth-limiting term [117]. Chap. 8 treats the adversarial half of verification — output that is wrong because someone made it wrong — and Chap. 12 the evaluation apparatus.

Authority is the set of powers actually granted to the execution layer: which tools, which data, which counterparties, which monetary limits, which irreversible actions. It is scarce in a precise sense: every grant of authority enlarges the blast radius, and blast radius is paid for in expected loss, which appears in Eq. 1.1 as $\mathbb{E}[L](a)$. Authority is also the locus of the accountability invariant. Because it cannot be transferred (Principle 1.3), it must be *bounded*, and bounded in a form a machine can enforce — hence the choke-point design of Listing 1.2. The empirical case for treating authority as the scarce resource rather than as configuration is now substantial: the documented failure taxonomy for agentic systems includes agent compromise, agent injection, impersonation and flow manipulation, and it has grown, not shrunk, with operational experience [156]; the corresponding practitioner risk list names tool misuse, identity and privilege abuse, memory and context poisoning, insecure inter-agent communication and cascading failures as top-ranked risks derived from real incidents [182]. Each of those is, structurally, an unauthorised expansion of authority. Chap. 8 is the systematic treatment.

Attention is the operator's finite capacity to notice, decide and intervene. It is the resource the form is designed to conserve and the one it most easily destroys. Raising autonomy reduces attention consumed per item but increases the consequence of each item that attention misses, and there is a well-documented perverse regime in which the human nominally supervising an automated process becomes progressively less able to supervise it precisely because the process usually works. Attention is also finite in a way the other two resources are not: verification can be capitalised into artefacts and authority can be enlarged by engineering, but a single operator's hours are fixed, and Sect. 1.9 shows that they are the denominator of the only productivity metric that matters. In a solo firm the approver duties that a larger organisation distributes across a compliance function, a reviewer and a manager all land on one person's attention, and there is no research establishing how many concurrent agents one person can meaningfully supervise at a given autonomy level. Chap. 9 treats attention as a scheduling and queueing problem with explicit capacity limits, and Chap. 6 treats its financial analogue, where the scarce resource is not the operator's minutes but the budget envelope that automated action consumes without asking.

Proposition 1.8. In a one-person AI company the binding constraint on scale is never model capability. It is, in order of typical severity, the availability of cheap verification for the domain, the safe bound on delegated authority, and the operator’s attention. A firm that cannot state which of the three currently binds does not know what to build next.

1.9 Revenue per operator hour, and how to read this book

A form defined by the concentration of accountability in one person needs a productivity metric with that person’s time in the denominator. Revenue per employee, the standard benchmark in the software industry, is the wrong metric for a firm with no employees: with a denominator of one it degenerates into revenue, and it silently rewards the accounting fiction of moving work to contractors. The metric used throughout this book is coarser and harder to game.

Model 1.9 (Revenue per operator hour). The governing productivity metric of a one-person AI company is

$$R_h = \frac{R}{H_{\text{year}}}, \quad (1.2)$$

where R is annual revenue recognised by the firm and H_{year} is the total hours the human principal works in the year — all hours, including specification, review, incident response, selling and administration, not billable or productive hours only. R_h has units of currency per operator hour and is the quantity that every architectural decision in this book is intended to increase.

The choice of denominator is deliberate and unforgiving. Counting all hours means that automation which shifts effort from production into supervision shows no gain, which is correct: the form’s value comes from work happening while the operator is not working, and any metric that permits supervision to be excluded would report success for a firm that had merely relabelled its labour. Counting recognised revenue rather than gross transaction value prevents resale-heavy structures from flattering themselves. The metric says nothing about margin, and must therefore always be read beside the unit-economics apparatus of Chap. 6: a firm can raise R_h by spending unboundedly on inference, and Eq. 1.2 will applaud while the bank balance falls.

Worked Calculation 1.2 Revenue per operator hour across the benchmark ladder

Inputs. Operator hours $H_{\text{year}} = 1,800$ (45 working weeks at 40 hours; a modelling assumption). Revenue points: the mean nonemployer business, from \$1.8 trillion of receipts across 30.4 million establishments [232]; the published median revenue per employee for private software companies, \$141,125 [208]; a reported average near \$300,000 per employee for listed software companies [194]; and the modelled target of Fig. 1.4, \$1,400,000.

Step 1 — the nonemployer baseline. $1.8 \times 10^{12} / 30.4 \times 10^6 = \$59,211$ of receipts per establishment. By Eq. 1.2, $R_h = 59,211 / 1,800 = \$32.90$ per operator hour. This is the statistical centre of the one-person economy, and it is roughly

what an hour of undifferentiated solo trading earns.

Step 2 — the software benchmarks. $R_h = 141,125/1,800 = \$78.40$ for the private-software median, and $300,000/1,800 = \$166.67$ for the listed-company average. A one-person firm reaching the private-software median per-person figure is already earning 2.4 times the nonemployer baseline per hour ($78.40/32.90 = 2.38$).

Step 3 — the modelled target. $R_h = 1,400,000/1,800 = \777.78 per operator hour. That is 9.9 times the private-software median per-person benchmark ($1,400,000/141,125 = 9.92$) and 23.6 times the nonemployer baseline.

Interpretation. The target is not a claim that solo firms typically earn this; it is a statement of what the architecture must accomplish. Each operator hour must carry roughly \$778 of revenue, which is only possible if the majority of delivered work is produced during hours the operator is not working. **Sensitivity.** If the operator works 2,200 hours instead of 1,800, the same revenue yields $1,400,000/2,200 = \$636.36$ per hour — an 18.2% fall in R_h for no change in revenue, which is the precise sense in which working longer is a productivity loss in this form.

The second use of Eq. 1.2 is as a feasibility test. Because review consumes the same hours that appear in the denominator, a revenue target, a price point and a review policy together imply a ceiling on how much of the work may be reviewed at all. That ceiling is frequently the fact that decides whether a business model is available to a solo operator.

Worked Calculation 1.3 The feasible review fraction implied by a revenue target

Inputs. Revenue target $R = \$1,400,000$; price per completed unit of delivered work \$70 (a modelling assumption — one subscription month, one delivered report, one processed case); operator hours $H = 1,800$; review fraction f , the share of units the operator inspects; review time $h = 6$ minutes per inspected unit.

Step 1 — units required. $N = R/70 = 1,400,000/70 = 20,000$ units per year.

Step 2 — review hours as a function of f . Review hours $= Nfh/60 = 20,000 \times f \times 0.1 = 2,000f$. At $f = 0.15$ this is 300 hours, or 16.7% of the operator's year — affordable. At $f = 0.40$ it is 800 hours, 44.4% of the year, leaving little for selling, building and incident response. At $f = 0.90$ it is 1,800 hours: the entire year is consumed by review and the firm produces nothing else.

Step 3 — the ceiling. Setting review hours equal to capacity gives $f_{\max} = H/(Nh/60) = 1,800/2,000 = 0.90$. Since specification, selling, finance and incident response must also fit inside H , the practical ceiling is far lower; reserving two thirds of the year for everything else leaves $600/2,000 = 0.30$.

Interpretation. At this price point and review speed, the business is feasible only if roughly 70% or more of units can be released without human inspection — that is, only at L3 or above on the ladder of Fig. 1.5, and only with automatic exception detection good enough that the un-inspected 70% is not a liability.

Sensitivity. Halving review time to $h = 3$ minutes doubles the affordable

fraction: 600 hours of capacity now covers $600 / (20,000 \times 0.05) = 0.60$ of units. Conversely, if the price per unit falls to \$35, then $N = 40,000$ and the same 600 hours cover only $600 / (40,000 \times 0.1) = 0.15$. Low price points do not merely compress margin; they mathematically require higher autonomy, and therefore stronger controls, than high price points do.

Worked Calculation 1.3 contains the strategic content of this chapter in compressed form. The choice of price point is simultaneously a choice of autonomy level, and therefore a choice of governance obligation. A solo firm selling a small number of expensive engagements can operate profitably at L1 or L2 and needs comparatively little control apparatus. A solo firm selling a large number of cheap units cannot survive at those levels arithmetically, and must therefore build the verification and containment machinery that L3 and above require before, not after, it acquires the volume. Choosing a business model without performing this calculation is the most common structural error in the form, and no amount of model capability corrects it.

The rest of the book follows the architecture of Fig. 1.2. Chap. 2 specifies the stack layers and the protocol surfaces between them — resolving the schematic’s five bands into six, because assurance (policy, evaluation and audit) is separated there from the operator-facing control plane with which the schematic combines it — including the interoperability protocols that make third-party tools available without bespoke integration [55]. Chap. 3 through Chap. 6 take the four agent teams in turn: product and engineering, marketing and growth, revenue and support, and the financial control of automated spending. Chap. 7 and Chap. 8 treat the two externally imposed constraint systems, legal obligation and adversarial pressure, and Chap. 9 builds the human oversight layer as an engineered queue rather than as a good intention. Chap. 10 addresses what happens when the substrate fails, Chap. 11 examines documented cases with their selection biases stated, and Chap. 12 assembles a complete instrumented firm. Two references recur as governance scaffolding throughout: the risk-management framework that supplies the vocabulary of AI risk functions [166] and the management-system standard against which a small firm’s AI governance can be structured without pretending to enterprise scale [109].

A closing note on register. This book makes no claim that a one-person firm is superior to a firm with colleagues, nor that the form suits most people or most markets. It claims something narrower and more useful: that the form is now technically constructible, that its economics have an interior optimum which can be computed rather than guessed, and that the difference between a solo firm that compounds and one that quietly accumulates liability is the presence of a control plane. Everything that follows is the engineering of that control plane.

Key Takeaways

- Firm size responds to the *differential* between external and internal coordination cost, not its level; a technology that lowers both symmetrically leaves the boundary unmoved (Sect. 1.1, Proposition 1.1).
- Delegation gains are real but setting-dependent: –40% time on writing,

$\approx +15\%$ resolutions per hour in support with novices gaining most, -55.8% on a greenfield coding task not reproduced on experienced maintainers' own repositories, and quality *degradation* outside the capability frontier (Table 1.1).

- The one-person firm is the modal business form — 30.4 million nonemployer businesses, \$1.8 trillion of receipts, 78.4% of all United States businesses — with mean receipts near \$59,000, which is the baseline any leverage claim must beat (Sect. 1.3).
- The form is defined by concentration of accountability, majority automated execution, and control through a designed control plane (Definition 1.2); authority may be delegated, accountability may not (Principle 1.3).
- Workflows have enumerable execution paths and agents do not, which determines testability, cost variance and attributability, and therefore the maximum safe autonomy level (Definitions 1.5–1.6).
- Autonomy is a per-workflow design setting, not a model property; published scales anchor on different variables and there is no accepted cross-walk between them, nor any legal mapping from level to obligation (Table 1.2).
- The cost-minimising autonomy level is interior: under the parameters of Eq. 1.1 the total falls from 103 at L0 to 63 at L3, then rises to 105 at L5 — worse than manual operation (Table 1.3, Worked Calculation 1.1).
- Verification, authority and attention are the scarce resources; improving verification is the only intervention that relocates the autonomy optimum rather than merely moving along it (Sect. 1.8).
- Revenue per operator hour $R_h = R/H_{\text{year}}$ counts all operator hours, so a price point and a review policy jointly imply a minimum autonomy level (Eq. 1.2, Worked Calculations 1.2 and 1.3).

Review Questions

1. State the Coasean condition determining the boundary of the firm, and explain why a technology that reduces both external and internal coordination costs by the same proportion leaves optimal firm size unchanged.
2. Distinguish a workflow from an agent using the control-flow criterion, and explain why the distinction changes the *kind* of assurance evidence an operator can produce, not merely its quantity.
3. Explain what the jagged technological frontier is, why an operator's subjective experience of a delegated task is an unreliable signal of its correctness, and what structural feature of a solo firm makes this hazard worse than in a firm with staff.
4. Using Model 1.7, take $C_L = (100, 74, 52, 33, 18, 9)$, then $C_G = (2, 6, 12, 22, 38, 62)$, and $E[L] = (1, 2, 4, 8, 17, 34)$. Compute $C(a)$ for all six levels, identify the minimum, and state the marginal cost of the $L3 \rightarrow L4$ step.

5. Take the same C_L and C_G but suppose an improved containment design reduces every residual expected-loss term by half, to $(0.5, 1, 2, 4, 8.5, 17)$. Recompute the totals and state whether the cost-minimising level moves.
6. A solo firm targets \$600,000 of annual revenue at \$25 per completed unit, with $H_{\text{year}} = 1,700$ operator hours and 5 minutes of review per inspected unit. Compute the number of units, the review hours at $f = 0.25$, the share of the operator's year those hours consume, and f_{max} .
7. Using Eq. 1.2, compute R_{h} for a firm with \$420,000 of revenue and 1,900 operator hours, and then for the same firm after the operator adds 300 hours of unpaid overtime with no revenue change. State the percentage change in R_{h} and explain why the metric is constructed to record it as a loss.
8. Assign an autonomy level to each of the following and justify it in one sentence each: publishing a blog post to the firm's own site; issuing a refund of \$20; issuing a refund of \$20,000; signing a customer contract; deploying a schema migration to the production database.
9. Design the exception criteria for an L3 support workflow in a firm with a single operator. Specify what must escalate, the maximum tolerable escalation rate given 1,800 operator hours, and what evidence each escalation must carry.
10. Given Table 1.2, explain why this book's L4 and L5 both map to a single NVIDIA level, and what that implies for the security controls an operator must build before moving a workflow from L3 to L4.

Discussion and Design Prompts

1. Task-level productivity gains of 15–56% have been measured repeatedly, yet the evidence that they aggregate into firm-level output is thin. Argue either that the one-person firm is the organisational form most likely to capture such gains — because it has no internal coordination to dissipate them — or that it is the least likely, because it has no division of labour to absorb verification. Support the position taken with the specific settings of the studies in Table 1.1.
2. Principle 1.3 holds that accountability cannot be delegated, while the liability literature anticipates responsibility shifting toward integrators and developers as autonomy rises [118, 132]. Discuss whether a solo operator can reasonably rely on that shift, and what evidence a single-person firm would need to retain in order to invoke it.
3. Model 1.7 places the optimum at L3 with the parameters of Table 1.3. Identify the domain characteristics that would move the optimum to L1 and those that would move it to L4, and propose two measurements a working solo firm could take from its own telemetry to estimate its own C_G and $\mathbb{E}[L]$ terms rather than assuming them.