SCALED

A SCALED FIELD GUIDE

The loop-engineered agency

A field guide to rebuilding your business for the agentic era

By Simon Penson

SCALED · SCALED.CO.UK · SCALEDOS.COM

About the author

Simon Penson

Founder, Scaled

Simon Penson is a multi-exited founder, having built and sold a diverse range of digital businesses including a content affiliate business, a venture capital fund and a large agency group.

In doing so he has merged, sold and invested in more than 100 B2B businesses, and learned both what it takes to build those businesses to real scale and how to maximise exit value.

Through that journey he has learned the language of mergers and acquisitions, and developed a passion for helping founders and senior teams maximise the value they have created. He writes about that work at **scaledos.com**.

You can reach him directly at
simon.penson@scaled.co.uk
scaledos.com

100+

B2B businesses merged, sold or invested in

3

Businesses built and exited

20yrs

Building and scaling agencies

CONTENTS

What is inside

PREFACE

Why I am handing you this

If you were in the room, you have already had the argument. You watched me stand up and tell you that the ground your agency is built on has moved, that the metrics you have optimised for a decade no longer earn a multiple, and that the quiet extinction of the old agency model is already underway in boardrooms you will never sit in.

You watched the 8.2x land. You saw the fossils. That was the why, and forty-five minutes is about the right amount of time to spend on a why, because a why does not change your Monday. A how does.

This book is the how. It is the thing I could not fit on stage without turning a keynote into a lecture, and it is the thing that actually matters once the applause has died down and you are back at your desk with a P&L that still looks like the old world and a team that still bills by the hour.

I have written it because I have spent the last stretch of my career sitting in the boardrooms where these decisions get made or ducked, and I have watched a small number of agencies quietly rebuild themselves into something that gets valued like software while everyone around them argues about day rates. The difference between those two outcomes is not talent, and it is not luck.

It is architecture. It is a decision, made in a particular week, to stop selling labour and start engineering loops.

I should tell you where I am writing this from, because it shapes every page. I built and sold a group of content and performance agencies to IPG in 2016.

During the earn-out that business went from around a hundred and twenty-five people to more than two hundred and seventy, and we hit roughly twenty-three million in revenue and six and a half million of EBITDA. I know what it feels like to grow an agency by adding people, because I did it, and I did it well, and I would not do it that way again.

Since then I have angel-invested into more than a hundred B2B businesses through Haatch, I have moved deeper into later-stage private equity, and I run Scaled, a consultancy that exists to help B2B founders grow faster and with less

pain. I sit on a handful of boards.

And in the last twelve months I have watched the question the whole market is dancing around get asked to my face, on a live diligence call, by a lower mid-market PE house looking at a European content agency I am the incoming Chair of. They asked me one thing.

Not margin. Not client concentration. Not churn. They asked: how defensible is content, really, with AI?

Content is not dying. The old content agency is.

My answer to them is this entire book, applied to one live deal. Content is not dying. The old content agency is.

Rebuild it around automation and workflow, keep the senior judgment at the top, and the thing becomes margin-enhancing rather than margin-collapsing. Content will be needed more than ever.

It will simply not be made the way most people are making it today. That answer held on the call, and it holds in your business, and the rest of these pages are the map that turns it from a good line into a plan you can run.

A word on how to read this. It is long on purpose. I have tried to be genuinely useful rather than merely inspiring, which means I explain what a thing is before I tell you why it matters, and I tell you why it matters before I tell you how to do it, and then I try to hand you enough of the how that you could start this week without hiring anyone or buying anything.

Read it in order the first time, because the argument compounds the same way the loops do. After that, treat it as a reference and come back to whichever chapter matches the problem in front of you.

There is an appendix of templates at the back for the same reason. This is meant to be a working document, not a keepsake.

One last thing before we start. You are the last generation of agency leader who will hold both the old craft and the new tools at the same time.

Everyone who comes after you will only have the tools. That is the single most valuable position in this industry for the next decade, and it is only valuable inside a window that is closing. So let us not waste it.

END OF PREFACE

Why I am handing you this

DIAGNOSIS

Part 1:

The Reset

The line through the market, and which side you are standing on.

SCALED

CHAPTER 1

The line has already been drawn

Somewhere in the middle of the last two years, without any announcement and without most people noticing the exact moment it happened, a line got drawn through the agency market. It is a thin line and it is easy to miss, because on the surface the two sides look almost identical.

Same office. Same client logos on the wall. Same kind of work leaving the building. But the businesses on one side of that line are being valued like software, and the businesses on the other side are being valued like staffing companies, and the gap between those two valuations is not a few percentage points.

It is a multiple of five to ten times on the same revenue. I want to spend this chapter making absolutely sure you understand which side of the line you are currently standing on, because everything else in this book is about getting you across it, and you cannot cross a line you cannot see.

Let me describe the world below the line first, because it is the world almost every agency was built in, and it is probably still the world your economics live in today. It is a world priced in time. You sell hours, or you sell days, or you sell a blended day rate that pretends not to be hours but absolutely is.

Your central operating metrics are utilisation, blended day rate, headcount and hours billed, and every one of those metrics ties the amount of money you can make directly to the number of people you employ. When you win more work, you hire. When you lose work, you have too many people and your margin collapses, which is exactly what happened to a great many agencies whose EBITDA cratered in the years when demand got choppy.

This is a linear business. Value in equals value out, and the value is human, and human value resets to zero at the start of every single project. You learned something brilliant on the last engagement and then you started the next one from scratch, because the learning lived in a person's head and the person was busy on something else.

That is the world that is ending. Not ending badly, not ending violently, but ending the way things end in an extinction event, which is to say quietly, for reasons the incumbents never quite believe until it is too late.

5-10x

The valuation gap on identical revenue, either side of the line

Now the world above the line. It is priced on outcomes, or on something closer to a token model where the client pays for a result delivered per unit of resource rather than for a person's time. Its systems compound, meaning the work it did last month makes the work it does this month cheaper and better rather than starting the meter again.

Its teams are small and its propositions are sharp and fast-evolving. And crucially, it earns software-like multiples, because a buyer looking at it sees recurring revenue, defensible intellectual property, real product, and margins that expand as the business grows rather than margins that get eaten alive by the next round of hiring.

Same market. Same clients. Often the same founder. The only difference is the architecture underneath, and the architecture is a choice.

Here are the two worlds set side by side, so you can hold the whole contrast in one glance before we go deeper on why each line matters:

DIMENSION	BELOW THE LINE (THE WORLD ENDING)	ABOVE THE LINE (THE WORLD BEGINNING)
Priced on	Time - hours, days, blended day rate	Outcomes - result per unit of resource
Core metrics	Utilisation, day rate, headcount, hours billed	Recurring revenue, retention, revenue per head

DIMENSION	BELOW THE LINE (THE WORLD ENDING)	ABOVE THE LINE (THE WORLD BEGINNING)
Relationship to headcount	Revenue and cost rise together, in lockstep	Output rises while headcount stays flat
What happens to learning	Resets to zero at the start of every project	Compounds - every cycle starts smarter
Margin as you grow	Flatlines or shrinks, eaten by hiring	Expands, as systems absorb cost
How a buyer values it	Like a staffing company	Like software
Typical exit multiple	Low single digits on EBITDA	Five to ten times, often on revenue

Read that table as a set of consequences that all flow from the first row, because they do. The moment you price on time rather than outcome, every other line follows almost inevitably.

Pricing on time means your metrics have to be about time, which means utilisation and day rate and hours billed, because those are the only things you can measure when time is the product. Metrics about time tie your revenue directly to headcount, because the only way to sell more hours is to employ more people to work them, which is why revenue and cost rise in lockstep and why margin flatlines as you grow, since every new pound of revenue drags a new pound of salary behind it.

And a business whose value is its people is a business whose learning resets every time a person leaves or a project ends, because the learning was never anywhere but in the people, which is precisely what a buyer sees and prices as a staffing company. The whole descent from the first row to the last is a single logical chain, and the good news buried in that is the same chain runs in reverse.

Change the first decision, from pricing time to pricing outcome, and you start pulling every other line across the border with it. That is what the rest of this book does, line by line.

Here is the part that people find genuinely hard to accept, so I will say it plainly. Measuring the outcome is not enough. A great many agencies have already made the leap to talking about outcomes, to putting a result on the invoice instead of a timesheet, and they think that is the transformation.

It is not. It is the easy half of it. An agency that measures the outcome but does not change the machine underneath is running what I call an open loop.

It watches the result. It reports the result. It never moves in response to the result and it certainly never gets smarter because of the result. You can measure your way into a beautiful dashboard and still be a staffing company with better positioning.

The businesses that cross the line run a closed loop instead. They measure, they move, and the moving compounds, so that every cycle leaves the system a little more capable than the cycle before.

That distinction, between the open loop that only observes and the closed loop that observes and improves, is the entire thing this book is about, and it is why the word engineering matters. You are not decorating your existing agency with some AI. You are re-engineering the machine so that it learns.

Labour resets every time. Infrastructure compounds.

You already know a company that got this catastrophically wrong from the other direction, by the way, and you use it. Think about the tools you rely on to tell you your organic rankings dropped. They are superb at telling you what happened.

They have no idea what to do about it, and no capacity to do it for you, and no memory that improves the next time the same thing happens. That is an open loop sold as a product, and it is exactly the gap that the agencies who cross the line are going to fill.

The measurement layer is commoditised. The layer that measures, decides, acts and learns is not, and it is where all the value is going to pool.

I want to leave you with a single image to carry through the rest of Part One, because it does more work than any framework. When the asteroid hits, and in this industry the asteroid is agentic AI dropping the cost of execution through the floor, it does not kill everything. It clears the board.

Every niche the giants monopolised suddenly stands empty. Every process that used to require a room full of people suddenly requires a loop and a handful of humans with judgment.

The businesses that die are not the small ones. They are the ones that were big and slow and built for an atmosphere that no longer exists. And the businesses that inherit the cleared board are the ones that were small enough, and fast enough, and deliberately enough engineered to move into the space the moment it opened.

Extinction rewards two traits, and neither of them is size. It rewards being fast, and it rewards having built infrastructure that compounds while everyone else is still resetting their labour to zero at the start of every job.

You are standing on one side of a line. The next chapter is about finding out, honestly and without flinching, exactly where.

END OF CHAPTER 1

The line has already been drawn

CHAPTER 2

Read your own fossil record

Before you can build anything, you have to be honest about what you have got, and this is the chapter most agency owners want to skip because it is uncomfortable. It asks you to look at your own business the way a diligence team looks at it, which is not the way you look at it, and it asks you to find the places where your enterprise value is leaking out of the building every single day without you noticing.

I am going to make you do it anyway, because you cannot engineer a loop to fix a problem you refuse to name, and because the founders who do this exercise properly are the ones who end up genuinely surprised by how much latent value they were sitting on.

Start with the P&L, but read it differently than you normally do. When you look at your accounts, you probably read them for profit, for the number at the bottom, for whether it was a good month. A buyer does not read them that way at all.

A buyer reads your accounts for what they reveal about the shape of the business, and the shape is what determines the multiple. There are four questions a buyer asks, and I want you to sit with them in the raw form a diligence team would put them, before I unpack why each one matters:

The four questions a buyer is really asking

How much of this revenue arrives whether or not you win anything new?

Has your revenue per head moved in three years, or just tracked headcount?

Is your margin expanding as you grow, or flatlining as you hire?

If you disappeared for three months and told nobody, what would break?

Take them one at a time, because each one is a probe for a specific weakness that the labour model hides. The first, on recurring revenue, is really a question about how much of your business you have to win all over again every month.

For most agencies, the honest answer is that very little of the revenue is genuinely recurring, because a retainer that can be cancelled on thirty days' notice and that depends on a single client-side champion staying in their job is not recurring in any way a buyer will pay a premium for. It looks recurring on the invoice and behaves like project work in reality, and a buyer knows the difference.

The second, on revenue per head, is a probe for whether you have learned anything you managed to keep. If revenue per head has stayed flat over three years, it means you grew by adding people in lockstep with revenue, which is the mathematical signature of a business that resets its learning to zero every time and has to buy more human hours to do more work.

The third, on margin trajectory, is deliberately about the direction and not the level, because a buyer pays for margin that expands as the business grows, and a linear labour business shows the opposite: margin that flatlines or shrinks because every new pound of revenue drags a new pound of cost behind it.

And the fourth, on founder dependency, is the one founders hate most, and it is the one they should sit with longest, because if the honest answer to what breaks when you leave is most of it, then you do not own a business. You own a very demanding job that happens to have employees, and a buyer will discount it heavily for exactly that reason.

Now do the harder read, the one the numbers only hint at. Somewhere in your agency, there is a set of things you do brilliantly, repeatably, across many clients, and that you have never once thought of as an asset.

It is the way you scope a particular kind of project. It is the specific sequence your best people run when they onboard a new account and it goes well. It is the framework one of your directors sketches on a whiteboard in every pitch that always seems to land. It is the process your delivery team follows that means your version of a fairly ordinary service somehow comes out sharper than the competition's.

Every single one of those is refined intellectual property that you have developed across a hundred client engagements, and every single one of them currently lives in a human head, gets applied inconsistently depending on who

is in the room, and walks out of the door at six o'clock every evening and sometimes for good when that person resigns. That is your fossil record.

It is the accumulated learning of everything your agency has ever done well, and right now it is fossilised, buried in people and inconsistent in application, generating almost none of the compounding value it could. The whole project of loop engineering is the excavation and reanimation of that fossil record, turning buried, inconsistent, human-bound process into named, consistent, system-bound infrastructure that compounds.

A loop that runs on inputs anyone can get is a loop anyone can build.

There is a specific diagnostic question I want you to sit with, and I want you to answer it honestly rather than aspirationally, because it is the fastest way I know to tell whether a loop is even possible in your business. The question is this: what data do you uniquely hold that nobody else does?

Not data you could buy. Not data everyone in your category can see. Data that exists inside your business, generated by your work, that a competitor could not replicate without doing what you have done for as long as you have done it.

If you can answer that question quickly and specifically, you have proprietary input, and proprietary input is the fuel that makes a loop defensible, because a loop that runs on inputs anyone can get is a loop anyone can build. If you cannot answer it, that is not a reason to stop, but it is the first thing your diagnosis needs to fix, because a loop with no proprietary input is a commodity dressed up as an asset.

Most agencies, when they think hard, find they are sitting on far more proprietary data than they realised, buried in the same place as the fossilised process, in the outcomes and the patterns and the client-specific learning that they have simply never gathered up and pointed at anything.

Do not rush this chapter in your own business. The founders I watch cross the line successfully almost all describe the same experience, which is that the diagnosis was the moment the whole thing became real, because it was the

moment they stopped thinking about AI as a threat coming from outside and started seeing it as a way to finally capture value that had been leaking out of their own building for years.

You are not behind. You are sitting on a fossil record that most of your competitors have not even thought to look for. The rest of this book is about digging it up and bringing it to life.

2

END OF CHAPTER 2

Read your own fossil record

THE LOOP

Part 2:

The New Unit

What a loop actually is, and how you build your first one.

PART 2 OF 5

SCALED

CHAPTER 3

A loop is not a workflow

I need to slow right down here and be very precise about language, because the single most common mistake I see agencies make is to hear the word loop, translate it in their heads into a word they already know, and then build the wrong thing with real conviction. They hear loop and they think automation.

They hear loop and they think workflow. They hear loop and they think of some clever chain of tools that takes a task off a person's plate.

None of those is a loop, and the difference is not academic, because the whole reason a loop earns you a software multiple and a workflow does not comes down entirely to this distinction. So let me build it up carefully, one layer at a time, because if you get this right everything downstream gets easier and if you get it wrong you will spend a year automating tasks and wondering why your valuation has not moved.

Start with the simplest unit, the task. A task is a thing you do once. You write a brief. You pull a report.

You send an email. When it is done, it is done, and if you need it again tomorrow, you do the whole thing again from the start. There is no memory and there is no leverage.

Automating a task is genuinely useful and I am not going to tell you not to do it, but understand what you have when you have done it: you have made one thing faster, once, and the benefit stops the moment the automation stops. It does not compound. It does not get better. It saved you some time and that is the entire extent of its value.

Now the workflow. A workflow is a task, or a chain of tasks, that repeats. You do the same sequence every time a new client onboards, or every time a campaign goes live, or every time a report is due.

A workflow is a real step up from a task because it applies to many instances rather than one, and automating a workflow can save a very large amount of time across a year, which is why most of what gets sold as AI transformation in

agencies is really just workflow automation. But hold on to the crucial limitation, because it is where the value ceiling sits: a workflow repeats, but it does not improve.

It runs the same way on the hundredth client as it did on the first. It has no mechanism for getting smarter, because there is no feedback flowing back into it. It is a faster version of the same thing forever, and a faster version of the same thing forever is worth something, but it is not worth a software multiple, because a buyer looking at it sees efficiency, not compounding.

Before I give you the loop, here are all three units laid out together, because the distinction between them is the single most important idea in this book and it is worth being able to see the whole ladder at once:

You sell tasks today. You will sell loops next.

	TASK	WORKFLOW	LOOP
What it is	A thing done once	A task or chain that repeats	A workflow that repeats and improves
Applies to	One instance	Many instances	Many instances, plus its own history
Has memory	No	No	Yes - captures what happened
Gets better over time	No	No	Yes - feedback improves each cycle
What it delivers	Time saved, once	Time saved, repeatedly	Compounding capability
A buyer sees	Nothing	Efficiency	An appreciating asset

	TASK	WORKFLOW	LOOP
Earns a software multiple	No	No	Yes

Notice that the leap that changes everything is the final column, and notice specifically that the row which separates a loop from a workflow is the same row that flips the bottom line from no to yes. A workflow and a loop are identical on the first three rows.

They differ only in that a loop has memory and gets better, and those two properties are what turn efficiency into an appreciating asset, and an appreciating asset into a software multiple. Everything hangs on that one difference, so let me build it up properly.

The loop is the thing that changes the category. A loop is a workflow that repeats and improves, and the mechanism by which it improves is the part I need you to really understand, because it is the whole point and it is the part everyone skips.

A loop has five moving parts arranged in a circle. There are inputs, and the inputs feed the work, and the work produces an outcome, and here is the part that matters: the outcome produces learning, and the learning flows back around to become better inputs for the next cycle.

Inputs, work, outcome, learning, better inputs, and then round again, except that this time it starts from a smarter place than it did last time, and the time after that it is smarter still. The feedback is not a nice-to-have bolted onto the side of the workflow. The feedback is the entire reason the thing is called a loop rather than a line, and it is the entire reason it compounds.

A workflow is a line that runs from start to finish and stops. A loop is a circle that runs from start to finish and then feeds what it learned back into the start, so that the machine you have on cycle fifty is materially better than the machine you had on cycle one, without anyone having rebuilt it, because it rebuilt itself a little on every pass.

Sit with why that compounding is so valuable, because it is the economic heart of everything. In the old agency, learning lived in people, and people are a resource that resets. When a brilliant strategist leaves, their learning leaves with them, and you are back to a lower baseline.

When a new junior joins, they start from close to zero and you spend a year getting them useful. Labour resets every single time. But a loop does not reset. The learning is captured in the system, in the improving inputs, in the accumulated pattern of what worked and what did not, and it stays in the business whether or not any particular person stays in the business.

Labour resets. Infrastructure compounds. That is the new asymmetry, and it is the reason a loop-engineered agency pulls away from a labour-engineered agency a little more every quarter, because one of them is climbing a staircase and the other one is running up a down escalator, doing enormous work just to stay level.

There is a sentence I use with founders that seems to unlock it, so I will give it to you directly. You sell tasks today. You will sell loops next. Right now, when a client buys from you, they are buying a bundle of tasks and workflows executed by your people, and the thing they are really paying a premium for is the skill of those people.

In the loop-engineered world, when a client buys from you, they are buying the accumulated learning of an entire system that has run this exact kind of engagement a hundred times and got smarter on every one of them, and that is a fundamentally more valuable and more defensible thing to sell, because your best competitor cannot match it by hiring good people. They would have to run the loop for as long as you have run it, and they cannot buy that time.

This is why I keep insisting on the word engineering. You are not automating your agency. You are building a machine that learns, and the machine that learns is the asset that a buyer will pay a software multiple to own.

CHAPTER 4

Find the one loop worth building first

The instinct, once the loop idea has landed, is to try to loop everything at once. I understand the instinct and I want you to resist it completely, because it is the single most reliable way to fail at this. Agencies that try to build seven loops simultaneously build seven half-loops, none of which ever fully closes, and a half-closed loop is worse than no loop at all because it consumes real resource and returns none of the compounding benefit, since the compounding only starts once the loop is actually closed and the learning is actually flowing back around.

The discipline that separates the agencies who get this right from the ones who stall is almost boringly simple. Do not build seven loops. Build one. Build it all the way, fully closed, until it is genuinely compounding, and let that first loop teach you how to build every loop after it, because it will, and the second one will take a fraction of the time the first one took. This chapter is about how you choose the one.

Begin with what I call the keystroke audit, and take the name literally, because the point of it is to find where repeatable effort actually lives rather than where you assume it lives. Most owners, asked where their team spends its time, will give you an answer based on the org chart and the service list, and that answer will be wrong, because the org chart describes the work people are supposed to do and not the work they actually do.

The real repeatable effort, the stuff that is quietly eating your margin, tends to hide in the gaps between the named services, in the reformatting and the re-keying and the chasing and the version-controlling and the thing that always seems to need doing again from scratch even though you could swear you did it last month. So do not audit from the org chart.

Audit from the keystroke. Watch, or have your people log, where the repeatable effort genuinely goes over a couple of representative weeks, and be ruthless about capturing the invisible work as well as the visible work. What you are looking for is not the glamorous work. It is the work that is repeatable, that

recurs across clients, and that currently requires a human to do something a system could do, because that work is where your loop candidates are, and the best candidate is almost never the one you would have guessed.

Once you have your list of candidates, you have to score them, because you can only build one first and you want it to be the right one. I score every candidate on three dimensions and then I multiply them together rather than adding them, and the reason I multiply is deliberate, because a candidate that scores brilliantly on two dimensions and near zero on the third should fall to the bottom of the list, and multiplication punishes a weak dimension in a way that addition hides.

The first dimension is margin drag: how much of your profitability is this work currently eating, whether through the hours it consumes or the seniority it ties up or the errors it introduces when it is done inconsistently. The second dimension is repeatability: how often and how consistently this same work recurs, because a loop needs volume of repetition to have anything to learn from, and a beautifully important task that happens twice a year will never accumulate enough cycles to compound.

The third dimension, and it is the one people most often underweight, is closeness to the client outcome: how directly this work connects to the result the client actually cares about and pays for, because a loop built around something adjacent to the client outcome saves you money, but a loop built around the client outcome itself becomes something you can reprice and resell, and that is a far larger prize. Margin drag, times repeatability, times closeness to the client outcome.

Score each candidate honestly, do the multiplication, and the winner is rarely the loudest option in the room. It is usually the quiet, repetitive, margin-eating thing sitting right next to the outcome the client cares most about, which everyone had stopped seeing precisely because it was so routine.

I want to give you the proof that this works, because it is easy to nod along and hard to believe until you see it applied. There is a link-building agency I point to whenever someone tells me their work is too bespoke and too human to loop, because on paper it was the least likely candidate you could imagine.

The work was highly manual. It depended on deep process knowledge held by experienced people. It was as close to the customer as it is possible to get. And that turned out to be exactly why it was the perfect place to start, because those same traits, manual and process-heavy and close to the outcome, are the precise signature of a high-scoring loop candidate.

The people in that business knew exactly what the end product had to look like, and that knowledge, which they had never once thought of as an asset, was the entire thing. They did not invent anything new. They took the process they already ran in their heads, named it, automated it, workflowed it, and closed the loop around it, and the process became the product.

They named what they already knew, and in naming it they turned a manual service into a compounding, defensible machine. That is the whole move, and it is available to you in whichever corner of your business scores highest on the three dimensions, which is why the audit and the scoring matter so much. Get the first loop right and everything after it gets easier. Get it wrong and you will conclude, incorrectly, that loops do not work in your kind of agency.

END OF CHAPTER 4

Find the one loop worth building first

CHAPTER 5

Build it in a loop

Here is the part where I hand you the actual method, and there is a deliberate joke buried in it that also happens to be the most important instruction in the book: you build a loop by running a loop. The process of engineering the machine that learns is itself a machine that learns, which means you do not need to get it perfect on the first pass, because the process is designed to improve on every pass, and the first loop you build teaches you how to build the second one faster and the third one faster still.

There are five stages, and I have deliberately named them so that they double as the master framework that runs your whole business later, which is why you will meet these same five verbs again in Part Four when we talk about enterprise value. For now, learn them as the build method.

Diagnose, Design, Deploy, Track, Compound. Here is the whole method on one page, including the specific way each stage most often gets killed, because knowing the failure mode in advance is how you avoid it:

STAGE	WHAT YOU ACTUALLY DO	THE "DO THIS"	HOW IT USUALLY DIES
Diagnose	Shadow the work until you see the real workflow, not the documented one	Sit with your best operator and capture every judgment call and workaround	Building around the tidy documented process, which fails the moment it meets reality
Design	Spec one loop around one high-value job, with the feedback built in	Answer explicitly how learning is captured and fed back	Designing a workflow by mistake - inputs, work, output, and no feedback
Deploy	Build against live systems and real client work	Deploy inside your best client, invest real capital, not spare time	Deploying into a sandbox, which surfaces none of the real exceptions

STAGE	WHAT YOU ACTUALLY DO	THE "DO THIS"	HOW IT USUALLY DIES
Track	Prove it with the people who do the job - time, cost, quality	Measure against the honest pre-loop baseline	Skipping it, or making it vanity, so the loop has no fuel to learn from
Compound	Ship it, keep the learning, point the pod at the next workflow	Write down what the system learned, not just what it delivered	Declaring victory at deploy and never reaching this stage at all

I will take each one in turn and go deep, because this is the operational heart of the book and the place where good intentions most often die for lack of a specific next step.

Diagnose comes first, and it is not the same as the audit you did in the last chapter, so do not conflate them. The audit told you which loop to build. The diagnosis tells you how the work you have chosen to loop actually gets done, and the critical word in that sentence is actually, because there is always a gap between the documented process and the real one.

The documented process is the tidy version that lives in an onboarding deck. The real process is the messy, exception-riddled, undocumented thing your best person does on instinct, with all the little judgment calls and workarounds and shortcuts that never made it into any deck because nobody ever thought to write them down.

If you build your loop around the documented process, you will build a loop that fails the moment it meets reality, because reality is made of the exceptions. So diagnose by shadowing. Sit with the person, or the people, who do this work best, and watch them do it, and keep watching until you can see the real workflow rather than the official one.

Capture the judgment calls especially, because the judgment calls are where the proprietary value hides, and they are the thing you will most need to decide whether to encode into the system or keep in human hands. The output of a good diagnosis is a genuinely honest map of how the chosen work really gets done, exceptions and all, and if that map surprises you, which it usually does, that is a sign you are doing it properly.

Build against real work, inside your best client, or you are building a demo.

Design comes second, and this is where you decide what the loop is actually going to be, which is a more consequential decision than it sounds. The discipline of design is to pick one high-value, repeatable job out of the diagnosis and spec the loop around it, and the word one is doing heavy lifting again, because the temptation at this stage is to design a loop that handles every variation and every edge case, and a loop that tries to handle everything on day one never ships.

Spec the core case, the version that covers the bulk of the volume, and design deliberately for the feedback from the start, because this is the step where most people accidentally design a workflow instead of a loop. Ask yourself, at design time, the question that separates the two: am I building something that automates a task, or am I building something that compounds?

A workflow design specifies inputs, work and output, and stops there. A loop design specifies inputs, work, output, and then the two pieces that make it a loop rather than a line, which are the learning, meaning how the system captures what happened and whether it was good, and the feedback, meaning how that learning flows back to improve the inputs on the next cycle.

If your design does not have an explicit answer for how learning gets captured and how it feeds back, you have designed a workflow, and you should go back and design a loop.

The other thing you do at design time is rewrite the offer, because a loop is not just an internal efficiency, it is a proposition, and the proposition should be written around the outcome the loop produces rather than the inputs it consumes. This is the point where you stop selling the ingredients and start selling the meal.

Deploy comes third, and I have watched more loops die at deployment than at any other stage, almost always for the same reason, which is that people deploy into a sandbox instead of into the real business. A loop built and tested against pretend data and imaginary clients tells you almost nothing, because

the whole value of the loop is in how it handles real inputs and real exceptions and the real messiness of live client work, and none of that shows up in a sandbox.

So deploy against live systems and real client work, inside your best client, the one who trusts you enough to be a partner in the build rather than a customer who expects perfection on day one. Deploying into your best client does two things at once.

It gives you real conditions to build against, which is the only environment that will surface the exceptions your diagnosis missed, and it gives you a proof point, because when the loop works inside your best client it becomes the story you tell in every pitch after that. Invest real capital at this stage rather than treating it as a side project someone runs in their spare time, because a loop built in the gaps between billable work will be built in the gaps forever.

Hire or borrow the product and engineering capability you need, put actual money on the table, and treat the build with the seriousness you would treat a new service line, because that is what it is.

Track comes fourth, and it is the stage that people either skip or turn into vanity, and both failures are fatal in the same way. You have to prove the loop works, and you have to prove it with the people who actually do the job, not with a slide that shows a hypothetical saving.

Measure the things that matter, which are time, cost and quality, and measure them against the honest baseline of how the work was done before, because a saving you cannot evidence is a saving a buyer will not pay for and a saving your own team will not believe. Tracking is not only about proving the case to the outside world, though, and this is the part people miss.

Tracking is also the mechanism by which the loop learns, because the feedback that flows back around to improve the next cycle is made of exactly this measured data about what happened and whether it was good. A loop that is not tracked cannot compound, because it has nothing to feed back.

So build the tracking in as part of the loop rather than bolting it on afterwards, and treat the numbers it produces as both the proof and the fuel.

Compound comes fifth, and it is the stage that turns a clever build into a durable asset, and it is also the stage that most agencies never reach because they declare victory at deploy and move on. To compound, you ship the loop,

you keep the learning, and you point the pod at the next workflow, and every part of that sentence matters.

Shipping means the loop is genuinely in production, running on real work, not sitting in a perpetual pilot that never quite goes live. Keeping the learning means you deliberately capture what the system learned, not just the result it produced, and this is worth dwelling on because it is the difference between a moat and a memory.

After every cycle, and certainly after every campaign the loop touches, write down what the system learned rather than only what it delivered. The result is a fact about the past.

The learning is an asset for the future, and that document, the accumulated record of what the system now knows that it did not know before, is your moat, because it is the thing a competitor cannot replicate without running the loop for as long as you have run it.

And pointing the pod at the next workflow is how the whole thing scales, because the first loop taught the pod how to build a loop, so the second loop is faster and the third is faster still, and you are now a business that builds compounding infrastructure as a repeatable capability rather than a business that ran one clever project once.

Notice the order, because the order is not arbitrary and getting it wrong is a common and expensive mistake. Margin expands first, because the loop takes cost out. Then the intellectual property gets named, because in building and tracking the loop you have finally articulated and captured the process you always ran but never owned.

Then your pitch win rate moves, because you now walk into pitches with a proof point and a named, defensible system rather than a promise about how good your people are. Margin, then named IP, then win rate, in that order, every time.

And underneath all five stages sits the same simple truth that makes the whole method work: one pod, one workflow at a time, each one faster to build than the last. You do not transform your agency in a single heroic push. You compound it, one closed loop at a time, and the compounding is the point.

PEOPLE

Part 3:

The New Organisation

The pod of two, the outcome org chart, and what only humans do.

PART 3 OF 5

SCALED

CHAPTER 6

The pod of two

The moment you have built one loop and watched it work, the question that follows is a hiring question, and almost everybody answers it wrong. They conclude, reasonably enough, that if loops are the future then they need a team to build them, and they start drawing up a job spec for a head of AI, or they start pricing an agency to come and do it for them, or they freeze entirely because the idea of building an internal capability from scratch feels like a project the size of the business itself.

It is none of those things. You do not need an AI team. You need a pod of two people, and you very probably already employ at least one of them.

The pod is made of two roles, and the roles are genuinely different, which is why it takes two people rather than one heroic individual. The first role is your best operator, and by best operator I mean the person who knows how the work really gets done, not the person with the most senior title.

This is the person from your diagnosis, the one you shadowed, the one whose instinctive judgment calls and undocumented workarounds are the actual value in the process. They do not need to know anything about AI.

Their entire contribution is that they hold the ground truth of the work, the real version rather than the documented version, and they can tell instantly when a loop is producing something that is subtly wrong in a way that a person who does not do the job would never catch. The second role is the loop builder, and this is the person who knows how to turn a process into a machine.

They are technical, but not necessarily a deep engineer, because most first loops are built on top of tools that already exist rather than from raw code. What the loop builder brings is the ability to look at the operator's real workflow and see where the automation goes, where the model sits, where the human judgment has to stay, and how the feedback gets captured and fed back around.

The loop builder knows how to turn it into a machine. The operator knows what the machine has to produce. Neither can do it without the other, and that is the whole reason it is a pod and not a hire.

The way the two of them work is a conversation, and I mean that literally, because every loop in your future org chart starts life as a conversation between these two people. The operator describes what they actually do, in detail, including the parts they have never articulated because they never had to.

The loop builder asks the questions that surface the hidden judgment, the where-do-you-get-that-from and the how-do-you-know-when-it-is-wrong questions, and translates the answers into a design. They build a first version, the operator uses it on real work and tells the builder every place it fell short, and the builder improves it, and that cycle repeats until the loop is good enough to deploy.

If that sounds exactly like the Diagnose and Design stages from Chapter 5, that is because it is, and it is not a coincidence, because the pod is the human engine that runs the build loop. The build method and the team structure are the same shape viewed from two angles.

70 to 9

What a rebuilt delivery organisation looks like on the other side

Practically, this means your first move after your first loop is not a recruitment drive. It is to identify your best operator, who is already on the payroll, and to pair them with a loop builder, who you may already employ and can second into the pod, or who you can bring in on contract for the first build while you decide whether the capability should become permanent.

The pod builds the loop, and here is the division of ownership that matters and that you must be explicit about from day one: the pod builds the loop, but the loop owner keeps the outcome. The pod is a building capability.

It constructs the machine and then it moves on to construct the next one. Ownership of the outcome the loop produces stays with the senior person accountable for that client result, which is the subject of the next chapter,

because it is the thing that reshapes your entire org chart.

END OF CHAPTER 6

The pod of two

CHAPTER 7

Rebuild the org chart around outcomes

The old agency org chart is a pyramid of people stacked on people, and it leaks value at every layer. Once you understand why it leaks, you cannot unsee it. At the top sits a managing director. Below them, a layer of account directors. Below them, account managers.

Below them, executives, and below the executives, often, a further layer of juniors, analysts, reporters, traffic and resource. The theory of the pyramid is that work flows down to the cheapest capable hands and judgment flows up to the most expensive.

It is a sound theory for a labour business, but it has a fatal property in the agentic era, which is that value leaks at every single layer, because at every layer a human is doing something a loop could do.

Every one of those humans resets to zero at the start of every project and carries none of the accumulated learning forward. The pyramid is a machine for converting learning into salary cost and then throwing the learning away.

The loop-engineered org chart is not a smaller pyramid. It is a different shape entirely, and the shape is this: senior people own outcomes, and loops own execution. At the top you have a small number of senior humans, each one owning the outcome for a portfolio of clients.

Beneath each of them, where the pyramid used to have three or four layers of people, you have loops, with agents inside the loops doing the execution that the layers of people used to do. There is no middle.

The account managers and the delivery managers and the project managers and the junior executives and the analysts and the reporters have not been fired into the sea. They have been elevated into the loop layer, meaning the repeatable, executional part of what they did is now done by the loop, and the judgment part of what they did has moved up into the senior outcome-owner role.

You do not get flatter by cutting people. You get flatter by promoting judgment and codifying everything else. That distinction is the difference between a redundancy exercise that hollows out your business and a re-engineering that makes it dramatically more valuable.

Let me give you the actual shape, because you asked for specifics and this is the part founders most want to see drawn out. It is not a projection; it is the shape of a mid-size agency I work with that is running this today. Nine humans. Twenty loops. The output of what used to take seventy people. Here is the whole roster:

#	ROLE	WHAT THEY OWN	WHERE THEY SIT IN THE POD
1	Founder / CEO	Category, capital, the biggest bets	Above the pods
3	Client Directors	Outcomes for a portfolio of 10-15 clients each	The operator half of every pod
1	Head of Loops	The loop library; quality-gates every loop before it ships	Owns the standard across pods
2	Loop Builders	Building and improving loops where most needed	The technical half of every pod, rotating
1	Senior Craft Reviewer	Judging what the loops produce; holding the line on taste	The taste layer above output
1	Commercial + Ops Lead	Pricing, contracts, finance	Was five people, now one

That is nine humans. Take them role by role, because each one exists for a precise reason. The founder owns category, capital and the biggest bets. This is the one seat that never gets absorbed into a loop because it is pure portfolio judgment.

The three client directors each own the outcomes for a portfolio of ten to fifteen clients. They are the operator half of every pod, the people who hold the ground truth of the client work. This is why they are the most numerous role, because outcome ownership does not scale infinitely across one person.

The head of loops owns the loop library itself and quality-gates every loop before it ships. Someone has to be accountable for the standard of the machines, and without that role, you get a sprawl of inconsistent loops that nobody trusts.

The two loop builders are the technical half of every pod, and they rotate across the client directors, building and improving loops where they are most needed. This is efficient because building capability is portable in a way that outcome ownership is not.

The senior craft reviewer is the taste layer, whose entire job is to judge what the loops produce and hold the line on quality. A loop will happily produce competent mediocrity at scale unless a human with real taste is judging its output and sending the weak work back.

And the commercial and operations lead covers pricing, contracts and finance, doing what used to take five people, because those functions are among the most loopable in the whole business. Everything else, the whole executional middle, has gone into the loop layer. Same client base. Same output. Ten times the margin.

I want to be careful about the human reality of this, because it is the part that keeps good founders awake, and rightly. Restructuring around loops is not a euphemism for mass redundancy. If you treat it as one, you will destroy the trust and the culture that your best people stay for, and you will lose exactly the operators whose ground-truth knowledge the whole system depends on.

The move is to elevate, not to eliminate. Your best executives become loop owners and loop builders. Your best account people become client directors owning outcomes.

The people whose roles were purely executional get retrained into the loop layer or into the judgment roles. Yes, over time the total headcount comes down, but it comes down through the elevation of the people worth elevating and the natural attrition of roles that the loops have genuinely absorbed, not through a spreadsheet exercise done in a single quarter.

The old agency rationed its best people, giving the A players to the biggest accounts and staffing everything else with whoever was free. In the loop-engineered agency, there are no B and C players getting handed the smaller accounts, because every client now gets your best thinking, codified into the system and applied consistently.

That is not only a commercial upgrade; it is a genuinely better business to work in, because the work that made people miserable, the repetitive reformatting and re-keying and chasing, is the exact work the loops have taken.

7

END OF CHAPTER 7

Rebuild the org chart around outcomes

CHAPTER 8

What only humans do, and how to price it

If loops do the execution, the reasonable fear is that there is nothing left for the humans to do and nothing left for the client to pay a premium for. This fear is worth taking seriously because it is what stops a lot of founders from committing.

So let me answer it directly, first by being precise about what genuinely stays human, and then, because this is the part that actually determines your economics, by telling you how to price it. A thing that stays human but that you give away for free might as well have been automated.

Five things stay human, and they are not vague. Here they are with the pricing logic attached to each, because naming them is useless unless you also change how you charge for them:

WHAT STAYS HUMAN	WHAT IT IS	HOW YOU PRICE IT
Taste	Judging whether the output is good, not merely correct	Baked into the premium - it is why your work is worth more than the loop's raw output
Strategy	Deciding what the client should be trying to achieve at all	Priced as senior advisory, upstream of and separate from delivery
Trust	The relationship that survives a competitor's lower price	Priced as retained access, not per project
Judgement	The call when the decision is hard and the data does not settle it	The premium a client pays for seniority in the room
Kill or scale	The portfolio decision on which bets deserve more	Founder and director level; the value is in the whole picture

Price the judgment. Automate everything underneath it.

Now take each one in turn, because the reasoning behind why it stays human is what tells you how confidently you can price it. The first is taste, by which I mean the judgment of whether the thing the loop produced is actually good, not merely correct.

A loop can produce something that is technically right and creatively dead, and only a human with genuine taste can tell the difference. Taste is the reason the senior craft reviewer exists in the org chart.

The second is strategy, the decision about what the client should be trying to achieve in the first place, which is upstream of any loop. No loop can set this for itself because it requires understanding the client's business and market and ambition in a way that is inherently human and relational.

The third is trust, the relationship with the client that means they pick up the phone when it matters and stay with you when a competitor undercuts you on price. Trust is built between people over time and cannot be manufactured by a system.

The fourth is judgment, specifically the judgment offered in the room when a decision is genuinely hard and the data does not settle it. This is the thing a client is really buying when they buy seniority.

And the fifth, which is the most senior judgment of all, is knowing what to kill and what to scale. This is the portfolio-level decision about which loops and which clients and which bets deserve more and which deserve to be shut down. That decision sits with the founder and the client directors because it requires holding the whole picture at once.

Now the part that matters, because naming what stays human is useless if you keep pricing it like labour. When execution gets ten times cheaper, which it does, the instinct of a labour business is to pass that saving on, to compete on the newly lower cost. That instinct is a race to the bottom that ends with you having automated your own margin away. Do not do that.

The correct move is to stop pricing the execution at all, because the execution is now cheap and getting cheaper and there is no premium to be had there. It is to price the five human things instead, because they are the things that stay scarce precisely as everything around them gets abundant.

This is the whole logic of the repricing that Part Four is about, but the principle belongs here because it flows directly from what stays human. Your invoice should reflect the value of the taste, the strategy, the trust, the judgment and the portfolio decisions. It should treat the execution the loops produce as the near-free commodity it has become.

The agencies that win are not the ones who use loops to do the same work for less money. They are the ones who use loops to make execution nearly free and then charge a premium for the human judgment that decides what the execution should be and whether it is any good.

If execution gets ten times cheaper, the question is what clients still pay a premium for. The answer is someone to own the loop, the taste, the judgment and the system that keeps getting better. That someone is you, and that is what goes on the invoice.

END OF CHAPTER 8

What only humans do, and how to price it

VALUE

Part 4:

The New Economics

Naming your assets, pricing the outcome, and the number buyers read.

PART 4 OF 5

SCALED

CHAPTER 9

Name the gold you are already sitting on

Everything in Part Two was about building a loop, and everything in Part Three was about organising your people around loops. This chapter is about the thing that turns all of that internal machinery into external, defensible, saleable value, which is the naming of your intellectual property.

I said in Chapter 2 that you are sitting on a fossil record of refined process that currently lives in people and generates almost none of the value it could. I said the whole project was to excavate it.

This is the excavation, done deliberately. It is worth its own chapter because the act of naming is not cosmetic; it is the moment a buried process becomes a named asset that a buyer can understand, value and pay for.

Start from the recognition that you have already done the hard part without knowing it. Every repeatable process you run, every framework one of your directors reaches for in a pitch, every workflow you have refined across a hundred clients, is a product waiting to be named.

Most owners cannot see it, and the reason they cannot see it is not stupidity; it is proximity. They are standing too close. The process is so familiar, so much a part of the water they swim in, that it does not look like an asset. It just looks like the way things are done.

The gold is already inside the building. You simply have not named it yet. Until you name it, it cannot be sold, defended or valued, because a buyer cannot buy a thing that has no name and no boundary and no articulation.

The mechanism of naming is more concrete than it sounds. Take the highest-scoring process from your loop work, the one you have now diagnosed and built into a closed loop, and give it three things it has never had. Give it a name, a real one, a proper noun that turns your version of a service into a distinct and ownable thing rather than a generic category.

Give it a defined boundary, a clear articulation of what it does and does not do, what goes in and what comes out, so that it is a discrete thing rather than a fuzzy capability. And give it a documented method, the accumulated record of how it works and what it has learned, which is the moat we discussed in the Compound stage.

Once a process has a name, a boundary and a documented method, it has stopped being tacit knowledge in your team's heads and become a named product with defensible intellectual property around it. That transition is exactly what moves you from being valued as a staffing company to being valued as something closer to software.

The link-building agency from Chapter 4 is the proof again, viewed now through the economic lens rather than the build lens. What they actually did, in the end, was not a technical achievement; it was an act of naming.

They took a process they already ran, that lived in the heads of their most experienced people, that they had refined across every client they had ever served. They named it, bounded it, documented it and automated it, and in doing so they turned deep tacit process knowledge into a named product.

The knowledge was always the asset. It was simply unnamed, and therefore unsellable and undefendable, right up until the day they named it.

You do not invent anything new when you do this. You name what you already knew, and the naming is what unlocks the value.

CHAPTER 10

Stop pricing time, price the outcome

This is the repricing, and it is the single most consequential commercial decision you will make in the whole transition, because you can build every loop and name every asset and still leave almost all the value on the table if you keep sending invoices that are priced in time. So I want to make the argument for the repricing carefully, because it is counterintuitive and it frightens people, and then I want to give you the specific path from where you are to where you need to be.

Begin with the number that should reframe how you think about the whole opportunity. For every pound a client spends on software, they spend around six pounds on services. That ratio is the shape of the market, and it tells you where the money actually is, which is not in the software, it is in the services layer wrapped around it, the implementation and the strategy and the ongoing management that make the software actually deliver a result.

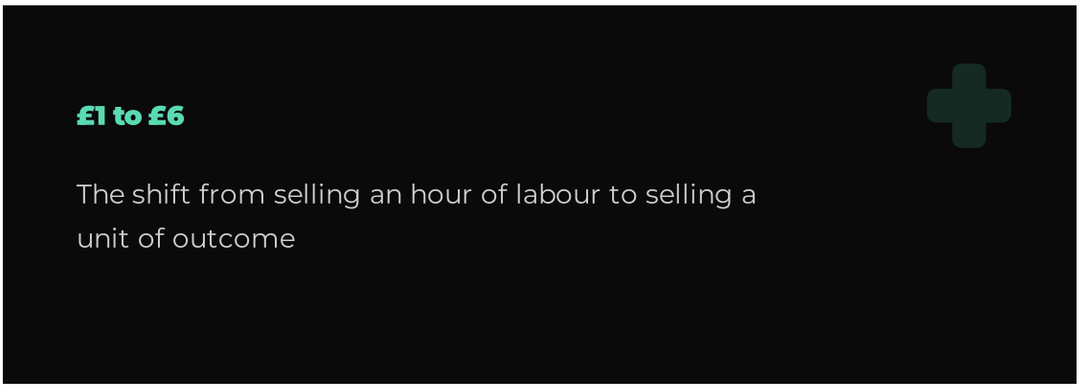
The lazy version of the AI story says that agentic software is going to capture that whole seven pounds, that the software will eat the services and the agencies will be left with nothing. That story is wrong about who wins, and it is wrong for a reason that should give you enormous confidence: software captures the outcome only if someone owns the last mile, the relationship and the category knowledge and the execution and the feedback loop and the taste, and that someone is the agency, not the software vendor, because the vendor sells a tool and the agency owns the implementation.

You are not being disintermediated by the software. You are the layer that makes the software worth paying for, and a loop-engineered agency is the thing that finally lets you capture a much larger share of that seven pounds, because you own both the tool and the last mile.

But you can only capture it if you reprice, and the repricing is a move from day rates to recurring revenue, which is precisely the shift that SaaS made twenty years ago when it moved the software industry off perpetual licences and onto subscriptions and rerated the entire sector in the process. The logic that worked for software works for you now, and it works because a loop, unlike a

person, produces a recurring outcome rather than a billable day, so it is naturally suited to being priced as a recurring subscription to that outcome rather than as a series of days sold.

The path is a progression, and you do not have to leap the whole way at once. Here are the four rungs of the ladder:



RUNG	YOU CHARGE FOR	WHAT IS BILLED	WHAT THE CLIENT IS BUYING
1	Hours	Time	Access to your people's time
2	Deliverables	Output	A defined thing produced
3	Expertise	Wisdom	A retainer for senior judgment
4	The loop	Outcome	A compounding result, priced as recurring revenue

Walk up the ladder and notice what changes at each rung. On the first rung clients paid you for hours, which was time billed, and the ceiling was hard because there are only so many hours and each one is finite and non-compounding. On the second rung the better agencies moved clients to paying for deliverables, which was output billed, a real improvement because it decoupled your fee from your timesheet, but still a fixed thing sold once.

On the third rung the best relationships paid for expertise, which was wisdom billed, the retainer for access to your senior judgment, and this is where a lot of good agencies have already climbed to and then stopped. The fourth rung is the one the loop makes possible and that almost nobody has reached, where

the client pays for the loop itself, for the compounding outcome the system produces, priced as recurring revenue against a result rather than a day rate against a person's time.

Each rung prices something more valuable and more defensible than the one below it, and the jump from the third rung to the fourth is the biggest of all, because it is the jump from selling a person's finite wisdom to selling a system's infinite and improving output. Day rates become annual recurring revenue. That is the sentence, and it is the same sentence the software industry wrote two decades ago, and it is your turn to write it now.

The practical move this quarter is to take one service, the one built around your first named loop, and rewrite its commercial model from a day rate or a project fee to a recurring subscription priced on the outcome it produces. You will find, when you do this, that you can charge more in total across the life of the relationship, not less, because you are pricing the compounding value of a system that gets better over time rather than the static value of a fixed number of days, and because the human judgment wrapped around the loop, the five things from Chapter 8, commands a premium that a day rate never captured. Stop pricing time. Price the outcome, and price it as the recurring, compounding, judgment-wrapped thing it now is.

10

END OF CHAPTER 10

Stop pricing time, price the outcome

CHAPTER 11

Build the number buyers read

Whether or not you ever intend to sell, you should run your business as though a buyer is watching. The metrics that make a business valuable to a buyer are the same metrics that make it a genuinely good business to own. The discipline of building the number a buyer reads forces you to fix the things that a labour business hides from itself.

This chapter is about the specific scoreboard that private equity actually reads, and about the trick of reverse-engineering your own destination from that scoreboard backwards. This is the most useful planning exercise I know.

Private equity has, in effect, one question now, and everything they look at is a way of answering it. The question is whether the business will be worth materially more in a few years than it is worth today, under their ownership, with a defensible reason for the increase.

They answer that question by reading six numbers, and I want you to start tracking all six from today. The eight-times conversations start with this exact page, and you cannot build a number you are not measuring. Here are the six, with what each one is really testing and what a loop-engineered business should be able to show:

METRIC	WHAT IT REALLY TESTS	WHAT GOOD LOOKS LIKE
Recurring revenue (% of total)	Whether you keep revenue by default or re-win it monthly	High and rising - the biggest lever on the multiple
Net revenue retention	Whether existing clients grow or shrink over time	Above 100% - clients spend more without new logos
Revenue per head	Whether you have escaped the linear labour trap	Rising sharply as loops absorb execution

METRIC	WHAT IT REALLY TESTS	WHAT GOOD LOOKS LIKE
Margin trajectory	Whether margin expands or flatlines as you grow	Expanding, as loops take out cost that used to scale
Growth rate	The numerator of every valuation	Fast, alongside expanding margin
Founder dependency	Key-person risk - what breaks if you leave	Low and falling, a side effect of the loop org chart

8.2x

The multiple the architecture earns, on the same revenue

Take each one in turn, because a founder who understands what the number is testing can move it deliberately rather than hoping. The first is recurring revenue as a percentage of total. Recurring revenue is the difference between a business that has to win itself again every month and a business that keeps its revenue by default. It is the single number that most separates a software multiple from a staffing multiple.

The second is net revenue retention, which is whether your existing clients spend more with you over time or less. A business whose clients naturally grow is a business that compounds without winning new logos. A loop-engineered agency should see this number climb as the loops make each client relationship more valuable.

The third is revenue per head, because it is the direct measure of whether you have escaped the linear labour trap. In a loop-engineered business, it should be rising sharply. This is because the whole point is to produce the output of seventy with the humans of nine.

The fourth is margin trajectory, not margin but its direction. A buyer pays for expanding margins, and a loop-engineered agency shows margin expanding as loops absorb cost that used to scale with revenue. The fifth is growth rate, straightforwardly. Growth is the numerator of every valuation, and a business growing quickly with expanding margins is the thing every buyer is hunting for.

And the sixth, the one founders least like to measure, is founder dependency. A business that cannot function without its founder is a business a buyer has to discount heavily for key-person risk. Reducing founder dependency, which the loop-engineered org chart does almost as a side effect, directly lifts the multiple.

Now, the exercise that ties the whole book together, which is to reverse-engineer your number from the destination backwards. You do not need a thirty-million-pound revenue business to reach a thirty-million-pound valuation. You need a three-and-a-half-million-pound revenue business valued like software.

The difference between those two routes is the entire argument of this book. The first route is the old linear grind of adding people until the revenue is enormous. The second route is the architectural one of building a smaller, loop-engineered, recurring, defensible business that earns a far higher multiple on far less revenue.

So, start from the destination. Fix the valuation you want as the destination. Work backwards to the architecture that earns it, which is loops plus named intellectual property plus recurring revenue. Work back one more step to the first loop, the single workflow you build this quarter, which is where the whole chain actually starts.

And then work back to Monday, to the diagnosis you run this week, because the thirty-million-pound number and the Monday diagnosis are the two ends of the same line, and the only thing that connects them is starting. Three years. Five moves. Starting today.

ACTION

Part 5:

The Decision

The window, and the first Monday of the rebuild.

PART 5 OF 5

SCALED

CHAPTER 12

The window, and the first Monday

I have given you the reset, the unit, the organisation and the economics, and if the argument has done its job then you already believe the destination is real and reachable. This final chapter is about the only thing left that matters, which is whether you actually start, and when, and I want to be honest with you about the window, because the window is the reason the when is not negotiable.

Every agency reinvents in this transition. Cutting edge or old school, early or late, there are no exceptions, because the ground has moved under all of them equally and standing still is simply the slowest way to reinvent. Being early to AI does not exempt you. It just means your rebuild starts from a better position, with more of the old craft still intact and more time to compound before the window closes.

And the window is closing, because you occupy a genuinely unique and temporary position: you are the last cohort of agency leaders who hold both the old craft and the new tools at the same time. The people who come after you will only ever have known the tools, and they will be faster with the tools than you, but they will not have the craft, the taste, the accumulated judgment about what good actually looks like that you built over twenty years in the old world.

That combination, old craft and new tools held in the same hands, is the most valuable position in this industry for the next decade, and it is only valuable inside the window, because once the window closes the craft advantage erodes and everyone is competing on tools alone. This is why there is no slow lane.

In an ordinary market, gradual is prudent. In an extinction event, gradual is a death sentence, because the businesses that move decisively inside the window inherit the cleared board and the businesses that move gradually are still deliberating when the board fills up with faster movers.

You do not have to build alone, and it is worth being clear about the three genuine routes through the window, because founders sometimes freeze thinking build-it-yourself is the only option. You can build, constructing your

own loops with your own pod, which is the route most of this book has described and the one that keeps the most value in your hands.

There is no slow lane.

You can partner, teaming with a SaaS business that has the tool but needs your execution and your last mile, which can be the fastest route into the new world if you find the right fit. Or you can combine, buying a business that gives you the capability or being bought by one that needs you, which is a legitimate and often overlooked path, particularly for founders who want the transition but not the multi-year build.

Every one of those three beats standing still, and the point is not which one you choose, the point is that you choose one and move, because the fatal option is the fourth one nobody names, which is to wait and see.

So let me give you the first Monday, concretely, because a decision that does not touch your calendar is not a decision. Reverse-engineer it from the destination the way we did in Chapter 11, but now turn it into a sequence you can actually start. The destination is your number, the valuation you decided you were building toward.

The architecture that earns it is loops plus named intellectual property plus recurring revenue. The first loop is one workflow, the highest-scoring candidate from your audit, built this quarter. And Monday is the diagnosis, the day you begin. Here is the ninety-day version, so there is no ambiguity about what start means. In the first thirty days you diagnose: you run the keystroke audit from Chapter 4, you score your candidates on margin drag times repeatability times closeness to the outcome, you pick the one, and you shadow your best operator until you can see the real workflow rather than the documented one.

In the second thirty days you design and begin to deploy: you spec the loop around the core case, you pair your operator with a loop builder to form the pod, and you build the first version against real work inside your best client. And in the third thirty days you track and begin to compound: you measure time, cost and quality against the honest baseline, you prove the case with the

people who do the job, and you write down what the system learned, not just what it delivered, which is the first entry in your moat. That is one closed loop in ninety days, and one closed loop teaches you how to build every loop after it.

There will be two agencies in your market in twenty-four months. They will have had the same tools, the same window, the same twenty-four months. One will sell for eight times revenue and the other will quietly close and never quite say why, and the only difference between them will be what they decided in the week they understood.

You are in that week now. This is that week, and this is that room. I have spent a career deciding where to put my time and my capital, and knowing everything I now know about where this is going, I back service to win, because service that owns the loop is the most defensible and most valuable thing in the whole chain. So build the loop. Start Monday. I will be watching to see which of the two agencies you become.

12

END OF CHAPTER 12

The window, and the first Monday

WORKSHEETS

Appendix

The templates

These are the working tools referenced throughout the book. They are deliberately simple, because the value is in filling them in honestly rather than in their sophistication. Print them, or copy them, and work through them in order. The first three map directly onto the method in Chapter 5, and the fourth is the scoreboard from Chapter 11.

TEMPLATE 1

The workflow scoring matrix

Use this to choose your first loop, following Chapter 4. List every repeatable process surfaced by your keystroke audit, then score each one from one to five on the three dimensions and multiply the three scores together. The highest total is your first loop. Multiply rather than add, so that a weakness on any single dimension pulls the candidate down as it should.

WORKFLOW / PROCESS	MARGIN DRAG (1-5)	REPEATABILITY (1-5)	CLOSENESS TO CLIENT OUTCOME (1-5)	TOTAL (MULTIPLY)

Margin drag: how much profitability this work currently consumes through hours, seniority tied up, or errors from inconsistency. Repeatability: how often and how consistently it recurs across clients. Closeness to outcome: how directly it connects to the result the client pays for. Winner: the highest total, which is rarely the loudest option in the room.

TEMPLATE 2

The loop blueprint

Use this to design your chosen loop, following the Design stage in Chapter 5. If you cannot complete the last two rows, you have designed a workflow, not a loop. Go back and design the feedback in.

ELEMENT	YOUR ANSWER
Loop name (the named asset)	
The one core job it does	
Proprietary input (what data you uniquely hold)	
Inputs (what goes in)	
Work (what the system does)	
Outcome (what comes out)	
Learning (how you capture what happened and whether it was good)	
Feedback (how the learning improves the next cycle's inputs)	
Human judgment retained (what stays with a person, and why)	
The best client to deploy inside first	

TEMPLATE 3

The build tracker

Use this to track and compound your loop, following the Track and Compound stages in Chapter 5. Fill the baseline in before you deploy, and revisit every cycle. The final column is your moat: what the system learned, not just what it delivered.

CYCLE	TIME (VS BASELINE)	COST (VS BASELINE)	QUALITY (VS BASELINE)	WHAT THE SYSTEM LEARNED THIS CYCLE
Baseline (before loop)				n/a
Cycle 1				
Cycle 2				
Cycle 3				
Cycle 4				

TEMPLATE 4

The metrics scoreboard

Use this to build the number buyers read, following Chapter 11. Record where each metric sits today, set the target that matches your destination valuation, and update quarterly. The eight-times conversations start with this page.

METRIC	TODAY	TARGET	DIRECTION THIS QUARTER
Recurring revenue (% of total)			
Net revenue retention (%)			
Revenue per head			
Margin trajectory (direction)			
Growth rate (%)			
Founder dependency (what breaks if you leave for 3 months)			

TEMPLATE 5

Reverse-engineer your number

Use this to connect the destination to Monday, following Chapters 11 and 12. Fill it in from the top down, then act on the bottom row this week.

STEP	YOUR ANSWER
The destination (valuation you are building toward)	
The architecture that earns it (loops + named IP + recurring revenue)	
The first loop (the one workflow you build this quarter)	
Monday (the diagnosis you run this week)	

A CLOSING NOTE

You have the argument, the method, the organisation, the economics and the templates. The only thing left is the decision, and the decision is not made in a document, it is made in a week, and the week is this one. Build the loop. Start Monday.

Build the loop. Start Monday.

SCALED

Remove the pain of growth.

SIMON PENSON · SCALED · SCALED.CO.UK · SCALEDOS.COM