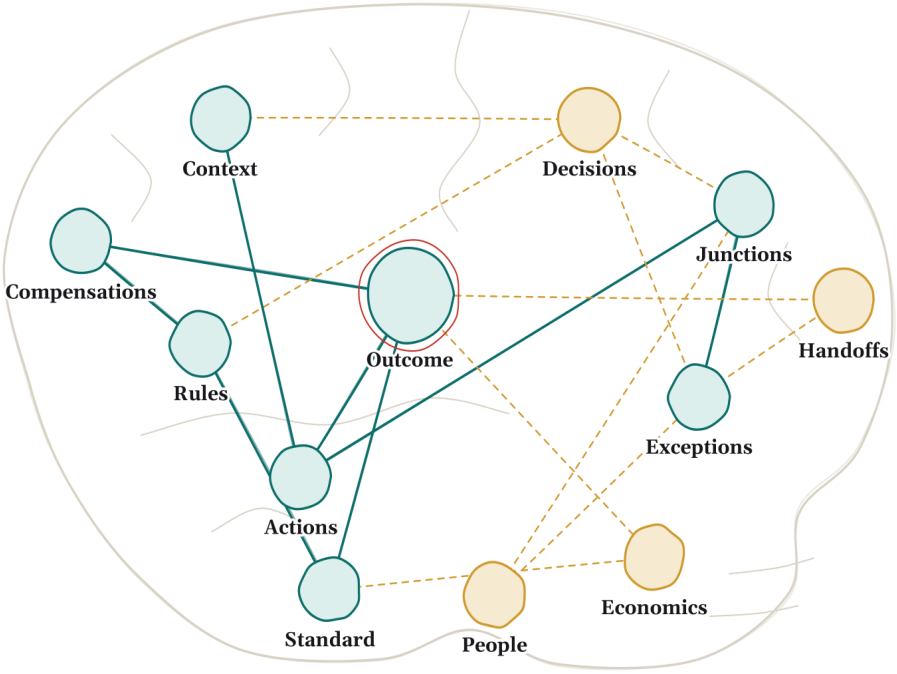
The Work Loop Canvas

How Work Is Redesigned When the Executor Decides



a century of designing work, one node at a time

1911	1913	1920s	1950s	1990	1992	2004	2010s	now
standard	*handoff*	*owner*	*stop cord*	*strip*	*rules*	*outcome*	*actions*	*decides*

> Strip the workarounds. Sort the decisions.
> Sign the loop.

Gennaro Cuofano

The Work Loop Canvas

How Work Is Redesigned When the Executor Decides

The Work Loop Canvas
How Work Is Redesigned When the Executor Decides

The Business Engineer Library · The Process Instrument

Gennaro Cuofano
businessengineer.ai

First published 2026. The composite firm in this book is illustrative; its figures are authored to teach the method and are not drawn from any company. Historical sources are listed in Notes and Sources.

Plates drawn for this volume in the Library's ink register.

Contents

Eleven Days 8

Introduction: The Hard Problem Moved 11

Part One: A Century of Designing Work **14**

I. The Line 16

II. The Divisions 19

III. The Stop Cord 21

IV. Obliterate, Then Encode 24

V. The Business Model on One Page 27

VI. The Bots 29

VII. From Business Model to Business Engineering 31

VIII. What Each Era Left Behind 34

Part Two: The Method Beneath the Canvas **36**

IX. Business Engineering, Beneath the Template 38

X. The Distribution You Operate In 40

XI. Abstraction, Compression and the Mental Model 42

XII. Back to the Real World 44

XIII. VTDF: Reading the Business Model 46

XIV. Where the Work Loop Plugs In 50

Part Three: The Inside-Out Problem **52**

XV. Outside-In, Inside-Out 54

XVI. Why AI Stalls at the Edge of the Core 56

XVII. The Unit Is the Work Loop 59

XVIII. Shaped Like the Organization's Brain 61

Part Four: The Canvas, Node by Node **63**

XIX. Outcome 65

XX. Compensations 68

XXI. Rules 71

XXII. Context 74

XXIII. Actions 76

XXIV. Decisions 78

XXV. Junctions 81

XXVI. Exceptions 83

XXVII. Handoffs 86

XXVIII. Standard 88

XXIX. People 91

XXX. Economics 93

Part Five: From One Loop to the Firm 96

XXXI. The Contested Decision 98

XXXII. The Work Loop Map 100

XXXIII. The Scale Gates 102

XXXIV. Running the Session 105

XXXV. What the Canvas Leaves Out 108

The Six Moves 110

The Meters 111

Epilogue: The Executor Changes 112

Appendix A: The Blank Canvas 113

Appendix B: The Worked Run, Filled 115

Appendix C: A Facilitator's Script 118

Notes and Sources 119

The Mental Models 121

The Thread: Shared Instruments Across the Library 123

Also in This Series 124

Also in this Library 125

A Note on How This Book Was Made

This book was written with an AI system working as a drafting partner, inside a method that sets the questions, the structure and the standard every chapter had to meet. The arguments, the instrument and the judgment calls are the author's. The drafting speed is the machine's. That arrangement is itself an example of the book's subject: a piece of knowledge work rebuilt as a loop, with a person at the junctions.

Contemporary companies appear by structure rather than by name in the chapters. Historical figures and published research are named, and the Notes and Sources section at the back gives full references. The insurer that runs through the book is a composite. Its numbers are illustrative, and its shape recurs often enough that the details change and the structure does not.

G.C.

Eleven Days

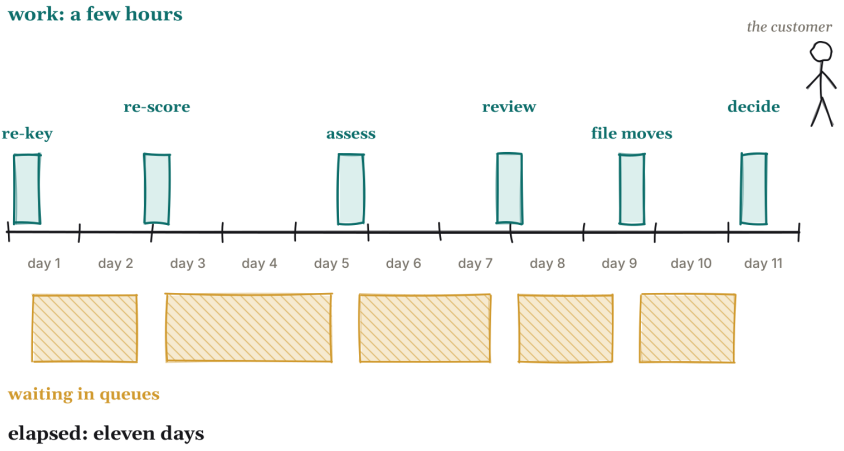
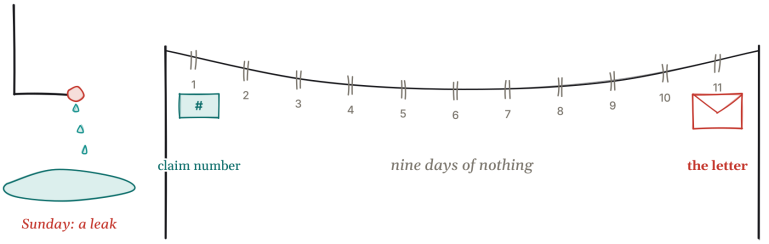


PLATE 1 Eleven Days

A homeowner notices water spreading across the kitchen floor on a Sunday evening. A pipe behind the dishwasher has failed. By Monday morning she has photographed the damage, called her insurer and been given a claim number. She is told someone will be in touch.

Eleven days later she receives a letter with a decision and an amount.

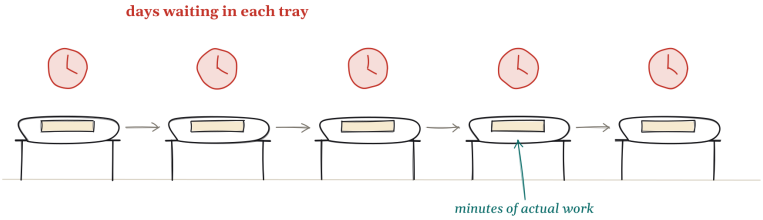


Like a washing line with eleven empty pegs: *the claim number arrives on the first day, and the letter with the answer only on the eleventh.*

Very little of those eleven days was work. Someone typed her call notes into a second system, because the call centre and the claims team use different software. A rating engine re-scored the claim overnight, because years ago it could only run at night. An adjuster assessed it. A second adjuster reviewed the assessment, because the first step once produced errors that it no longer

produces. A paper file moved between two teams on two floors. The claim waited in five queues.

None of this was anyone's mistake. Each step was added for a good reason, by a capable person, at a time when the reason was real. Over the years the process was measured, improved and documented. It is, by most standards, a well-run process.

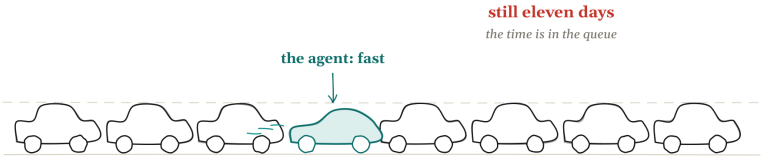


Like a letter passed along five desks: *each clerk handles it in minutes, but it lies in every in-tray for days, and nobody did anything wrong.*

Now the insurer wants to use AI. The obvious move is to take the process map and attach an agent to the steps that look automatable: the re-keying, the scoring, the first assessment. The agent will be fast. The claim will still take close to eleven days, because the time lives in the waits, the reviews and the handoffs that nobody questioned.

There is a second problem, and it is harder to see. Somewhere in those eleven days, people make decisions that nobody wrote down. Whether a borderline claim deserves a goodwill payment is one of them. Claims managers have paid them for years. The policy team says they were never authorized. Nobody noticed the disagreement, because experienced people quietly resolved it case by case. An agent cannot resolve it quietly. It will either guess or stop.

This book is about that gap. It is the gap between a business that knows what it sells and a business that knows how its work actually gets done. For most of the last twenty years the first kind of knowledge was the scarce one. Today the second kind is. The book offers one instrument for closing the gap, explains where each part of it comes from, and shows how to use it from a single process to the whole firm.



Like a sports car stuck in a traffic jam: *a faster engine changes nothing when the time is spent waiting in the queue, not driving.*

Introduction: The Hard Problem Moved

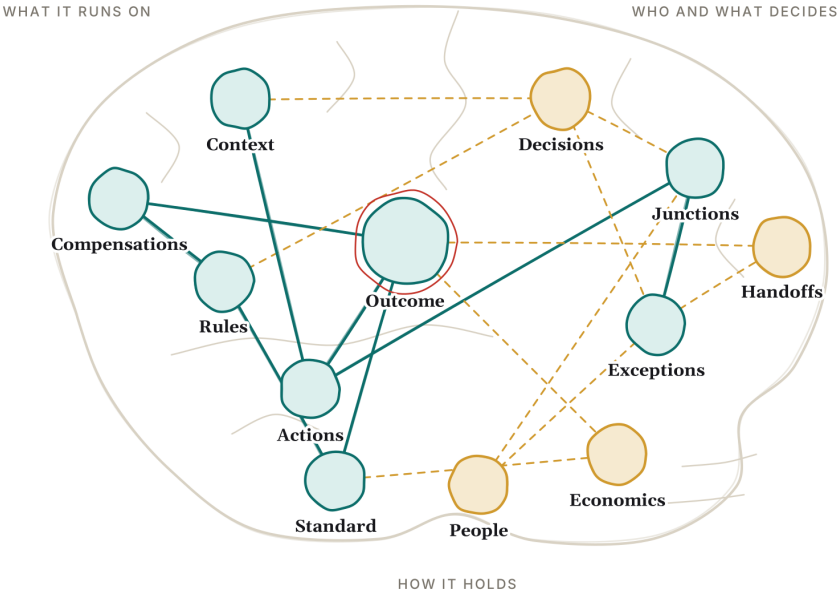


PLATE 2 The Work Loop Canvas

Every era of business leaves behind one picture that people draw on whiteboards. For the 2000s it was a sheet of paper divided into nine boxes. Founders filled it with sticky notes. Investors asked for it. Business schools taught it. It made a hard question easy to discuss: how does this company make money?

That question still matters, but it is no longer the hard one. A team can describe a workable business model in an afternoon, test it in a week and copy a competitor's in a month. The hard question has moved inside the firm. It asks how the work actually gets done, and whether AI can do part of it.

Most companies are struggling with that question. They run AI pilots at the edge of the business and cannot move them into the core. The pilots work. The core does not change. The reason is not the technology. It is that the core was never mapped in a form a new kind of worker could use.

The Work Loop Canvas is a one-page map of a single work loop. A work loop is one outcome, owned end to end, together with everything needed to produce it well. The canvas draws that loop as twelve connected nodes, shaped like a small brain, because the parts depend on each other the way regions of a brain do. Its companion, the **Work Loop Map**, places every loop in the firm on one page and draws the handoffs that wire them together. Five **Scale Gates** connect the two. They say how far a loop may go today, from a shadow trial to the core of the business.

How the book is organized

The book has five parts, and their order follows the way the problem unfolds.

Part One is history. It walks through a century of designing work, from Taylor's stopwatch and Ford's moving line to software robots. Each era left behind one part that a modern loop needs. Business engineering, which arrived with the platforms, is the turning point: it treats a business as a running system rather than a fixed design. None of the earlier schools had an executor that reads context and makes decisions. That is what changed.

Part Two is the method. Business engineering starts from the distribution of the domain a business operates in, abstracts and compresses what it sees into mental models, tests them against real cases and builds intuition that sends it back to the distribution. VTDF, the Business Engineer's way of reading a business model, is one product of that loop. The Work Loop Canvas is another, one level down.

Part Three is the argument. It explains why AI, unlike the web, has to work from the inside out, why pilots stall at the edge of the core, and why the unit of design has to change from the task to the loop.

Part Four is the instrument itself, one node per chapter, in the order a team should fill them. Each chapter gives the question, why it matters, how to fill the node, the test for when it is signed and the way it usually goes wrong. The composite insurer from the opening runs through every chapter.

Part Five moves from one loop to the firm. It covers the contested decision, the Work Loop Map, the Scale Gates and how to run a session. It ends with what the canvas deliberately leaves out.

The appendices hold the blank canvas, the fully filled worked run and a facilitator's script. The book is meant to be used, not only read. A team should be able to finish it and run its first session the next morning.

PART ONE

A Century of Designing Work

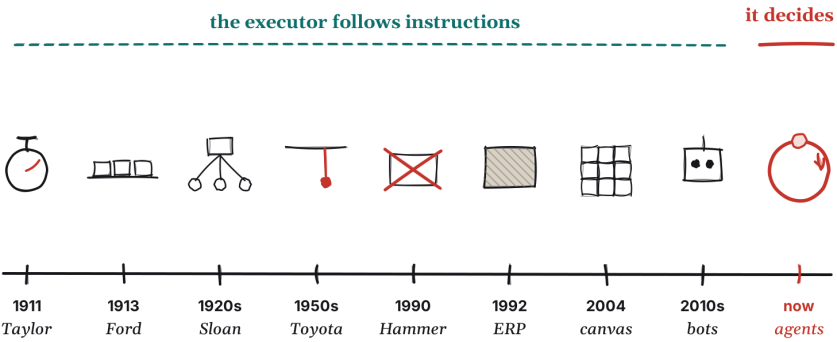


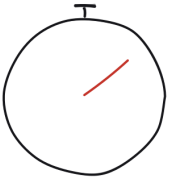
PLATE 3 A Century of Designing Work

Every generation of managers has redesigned work around the executor it had. The executor is whoever, or whatever, actually does the work. For most of the last century that executor followed instructions. Each new school of management found a better way to write those instructions down.

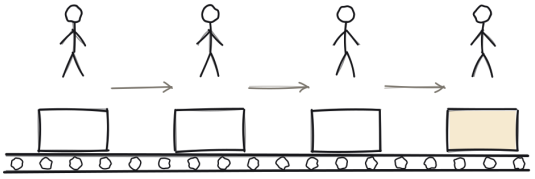
The history matters for a practical reason. Almost every idea in a modern process map comes from one of these schools. So do most of the habits that now slow AI down. A team that knows where each habit came from can tell which ones still serve the outcome and which ones are leftovers.

I

The Line



one best way, measured
Taylor, 1911



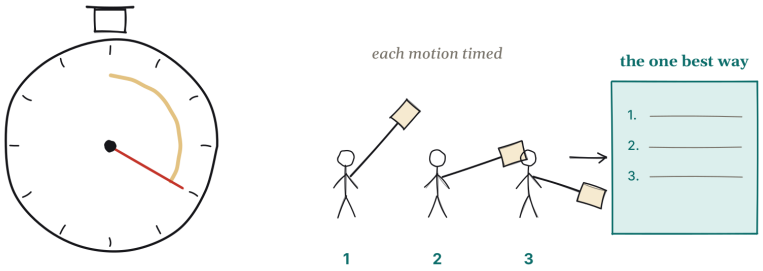
moving line, Highland Park, 1913



PLATE 4 1911–1913: The Line

In 1911 Frederick Winslow Taylor published *The Principles of Scientific Management*. His claim was simple. Every task has one best way, and management's job is to find it, measure it and teach it. Taylor timed workers with a stopwatch, broke each job into motions and wrote down the fastest sequence. Work became something that could be studied and written as a standard.

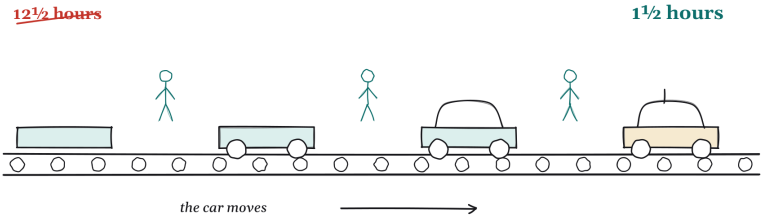
Taylor's methods were resented by many of the people they were applied to, and with reason. They treated the worker as a part to be tuned. But the underlying idea survived every backlash: a task can be described precisely enough that someone else can perform it to the same result. Every standard operating procedure written since descends from it.



Like a stopwatch held over a shoveller: *time every motion, keep the fastest sequence, and write it down so anyone can repeat it.*

Two years later Henry Ford put the idea in motion. In 1913 his Highland Park plant in Michigan introduced a moving assembly line for the Model T. Instead of workers moving around a stationary car, the car moved past stationary workers. Assembling a chassis had taken about twelve and a half hours. Within a few months it took about an hour and a half.

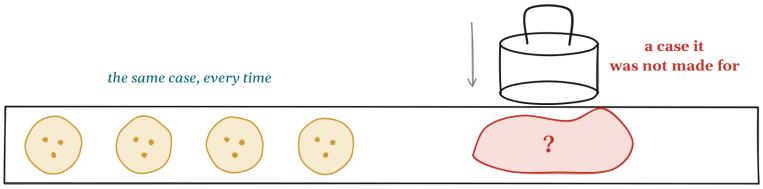
The line turned work into a designed flow. Each station received the car, did one thing and passed it on. The designer's job shifted from describing a task to arranging a sequence of tasks so the whole moved at a steady pace. **The handoff became the basic unit of industrial work**, and most process maps are still drawn that way, as boxes joined by arrows.



Like Ford's moving line: *the car rolls past workers who stay put, each does one thing and passes it on, and twelve hours become ninety minutes.*

Ford also discovered the cost of the design. The line was efficient because nobody on it needed to think. Turnover at Highland Park was so high that in 1914 Ford doubled pay to five dollars a day to keep people on it. The flow worked only as long as every case was the same case.

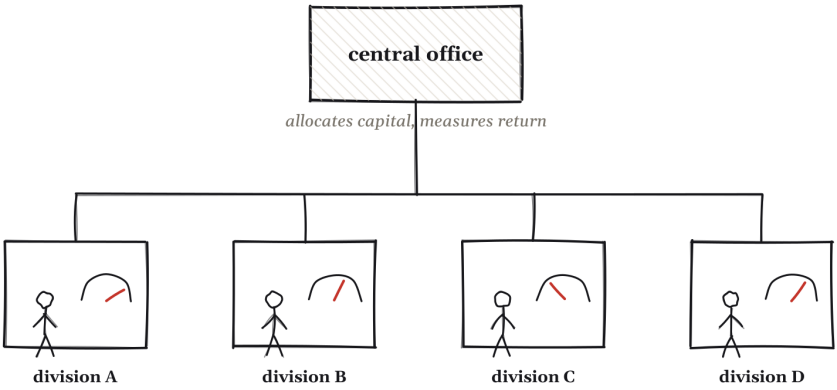
Two parts of the Work Loop Canvas come from this era. The standard, a written description of a good result, is Taylor's. The handoff, the point where work passes from one place to the next, is Ford's.



Like a biscuit cutter on a tray: *every biscuit comes out the same, until a lump of dough arrives that the cutter was never shaped for.*

II

The Divisions

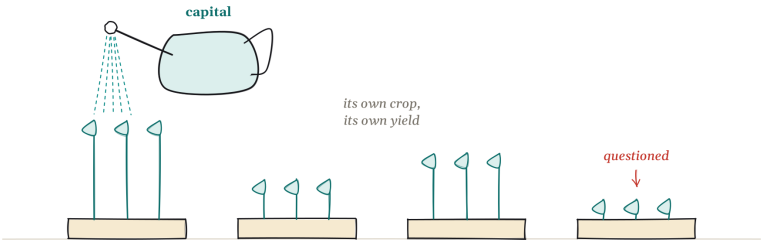


each with a named head and a return

PLATE 5 1920s: The Divisions

Ford's company ran on one product and one will. When the market wanted variety, it struggled to provide it. Alfred Sloan built something different at General Motors.

In the 1920s Sloan organized the company into divisions, each running its own operations for its own segment of the market. A central office did not run the divisions. It allocated capital among them and measured each one on its return. Divisions that earned their capital got more. Divisions that did not were questioned.

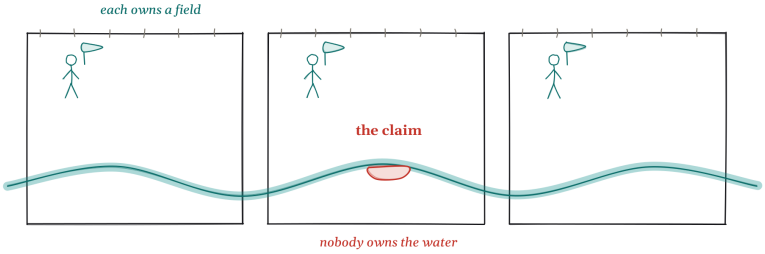


Like a gardener with one watering can and four beds: *each bed grows its own crop, and the water goes to the beds that yield the most.*

The historian Alfred Chandler later described this as the multidivisional form, and it became the standard shape of the large company for most of the century. Its real innovation was not the organization chart. It was accountability that could be read on a page. Each division had a named head and a number. A manager owned an outcome and answered for its economics.

That pairing of an owner with a measured result is the second inheritance a modern loop depends on. A loop without a named owner drifts, because nobody answers when it produces bad results. A loop without measured economics cannot be compared with anything, so nobody can say whether it is better than what it replaced.

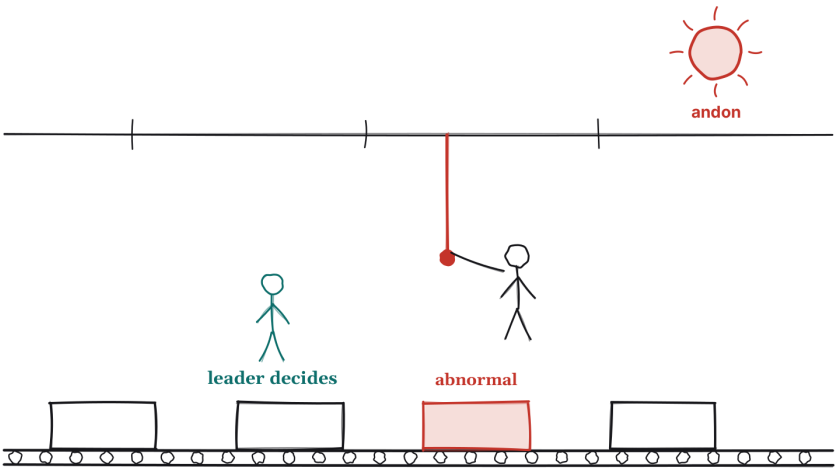
Sloan's model also carried a weakness the canvas has to correct. Accountability ran down the hierarchy, from the central office to division heads to their managers. Work, however, runs across the hierarchy, from one team to the next. A claim passes through intake, assessment and settlement, each in a different department with a different head. Everyone owns their piece. Nobody owns the claim. The work loop puts the owner where the work is, on the outcome rather than on the department.



Like a stream crossing three fenced farms: *each farmer answers for his own field, but nobody answers for the water that runs through them all.*

III

The Stop Cord

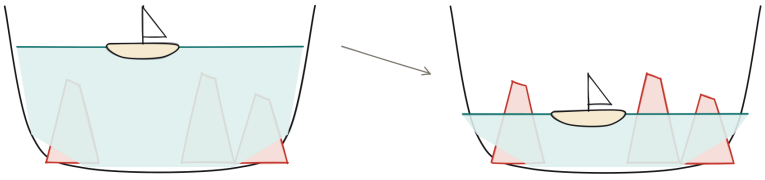


from the Toyoda loom that stopped when a thread broke

PLATE 6 1950s–1970s: The Stop Cord

After the Second World War Toyota faced a problem Ford never had. It needed to build many models in small volumes, with little capital and little space. Ford's line, built for one model at enormous scale, did not fit.

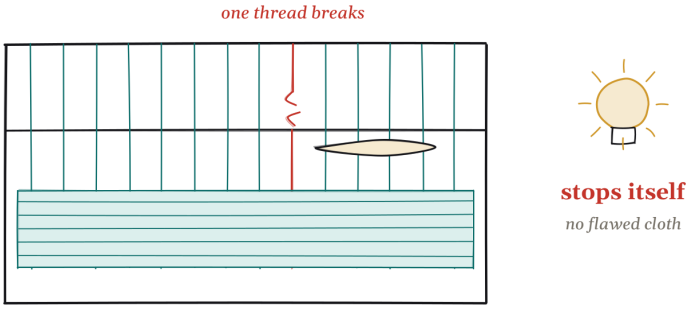
Taiichi Ohno and his colleagues built what became the Toyota Production System around two ideas. The first was just-in-time: make only what the next step needs, when it needs it. Inventory between steps was treated as waste, because it hid problems and tied up capital.



Like a lake that hides its rocks: *high water, the stock piled between steps, floats the boat over every problem; lower it and the rocks appear.*

The second idea was jidoka, sometimes translated as automation with a human touch. It came from Sakichi Toyoda's automatic loom, which stopped itself when a thread broke instead of weaving defective cloth. Toyota applied the same rule to people. Any worker who saw a problem could pull a cord running along the line, called the andon. A light came on, a team leader came over, and together they decided what to do. If they could not fix it quickly, the line stopped.

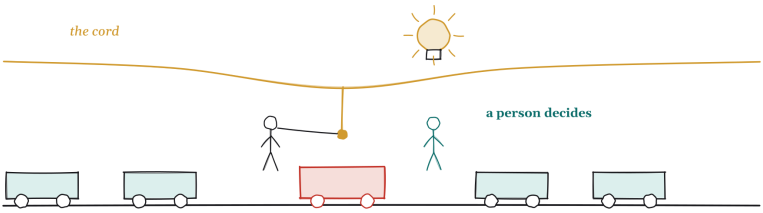
Western managers who visited were often shocked. Stopping the line was the worst thing that could happen in a Ford-style plant. At Toyota it was the point. A line that stops on every abnormality forces problems into the open, where they get solved instead of passed downstream.



Like Toyoda's automatic loom: *when a single thread snaps, the machine stops itself instead of weaving metres of flawed cloth.*

The andon cord is a junction in everything but name. The routine work runs on its own. An abnormal case stops it. A person with authority sees the evidence and decides. Toyota designed this decades before anyone talked about agents, and the principle carries over almost exactly. An agent should run the routine cases, stop on the abnormal ones and hand them to a person who can decide.

Toyota left two parts of the canvas. The junction is the cord. The exception queue is what happens after the cord is pulled: a defined place for the cases the routine cannot handle, worked by a named person against a time limit.



Like the andon cord on Toyota's line: *routine cars roll on, an odd one gets the cord pulled, a lamp lights, and a team leader decides.*

IV

Obliterate, Then Encode

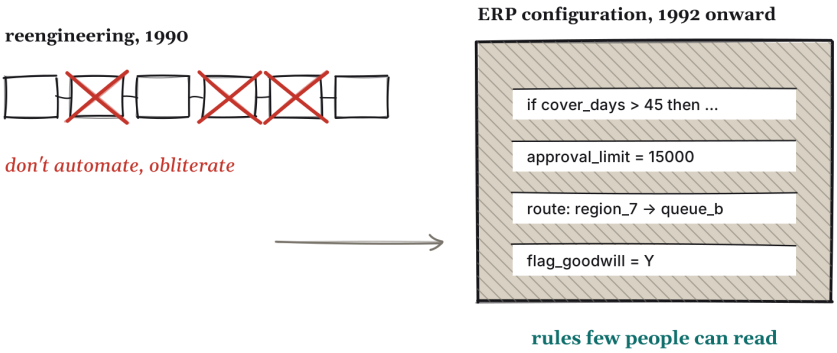
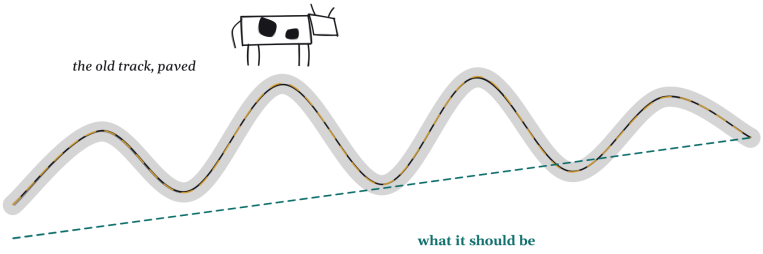


PLATE 7 1990s: Obliterate, Then Encode

By 1990 large companies had computers everywhere and little productivity to show for them. Michael Hammer explained why in a Harvard Business Review article titled "Reengineering Work: Don't Automate, Obliterate." His argument was that companies had used technology to speed up old processes instead of questioning them. They had paved the cow paths.

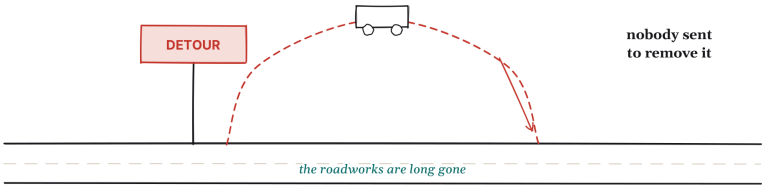
Hammer's examples were vivid. One large company had hundreds of people in accounts payable, matching purchase orders, receiving documents and invoices. A competitor did the same work with a fraction of the staff, because it had redesigned the process so that most of the matching was unnecessary. The lesson was that many steps existed only because of how work used to be done. They should be removed, not automated.



Like paving a cow path: *the tarmac makes the old winding track faster, but every bend stays, when the road should have been redrawn straight.*

Hammer and James Champy turned the argument into a book in 1993. Reengineering became one of the most popular management ideas of the decade, and then one of the most criticized. In practice it often failed. Projects were too large and too fast. The word became a polite label for layoffs, and employees learned to fear it.

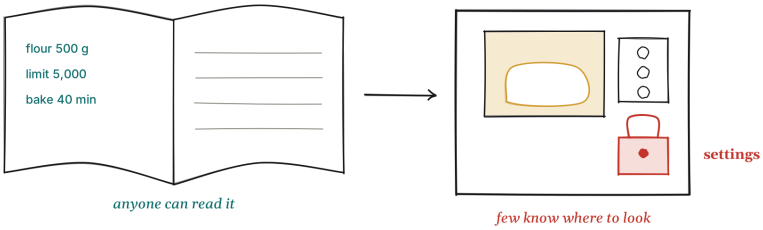
Reengineering had the right diagnosis and no executor. The redesigned processes still depended on people following new instructions and on systems that were hard to change. The diagnosis, though, is exactly the one the canvas starts from. Its second node, compensations, is Hammer's instruction in a sentence: strike every step that exists only to work around a constraint that no longer applies.



Like a detour sign left standing after the roadworks ended: *reengineering said take it down, but nobody was sent to do it, so cars still loop round.*

The same decade did something that made the next problem worse. SAP released R/3 in 1992, and enterprise resource planning spread through large companies. These systems wrote processes into software as configuration. Approval limits, routing rules and coverage conditions moved out of manuals and into settings screens.

That made processes more consistent. It also made their rules harder to see. A rule in a manual can be read by anyone. A rule in a configuration table can be read by the few people who know where to look. Decades later, when a company tries to tell an agent what the rules are, it often finds that nobody can list them. The canvas's rules node exists to bring them back into the open.



Like a recipe moved from the cookbook into a bread machine's program: *the loaves come out the same, but nobody can read the recipe any more.*

V

The Business Model on One Page

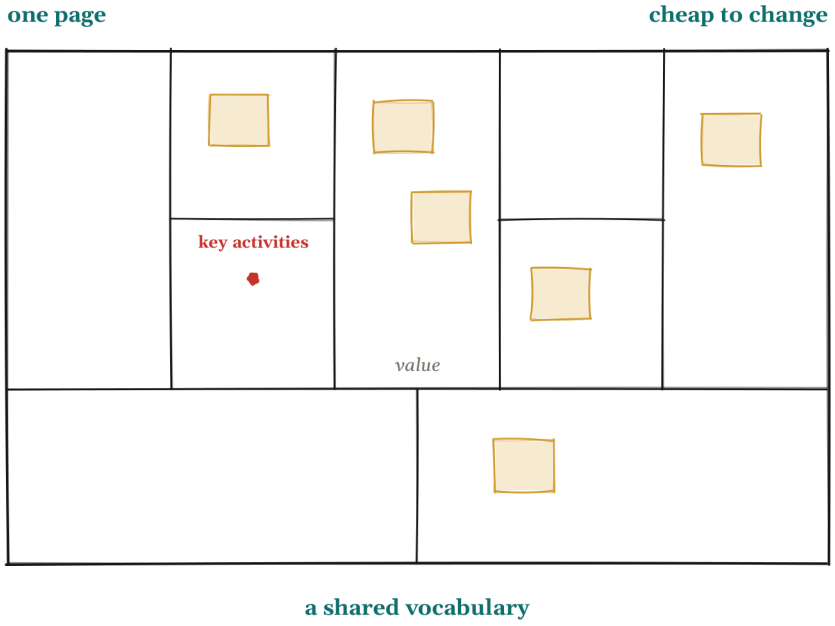
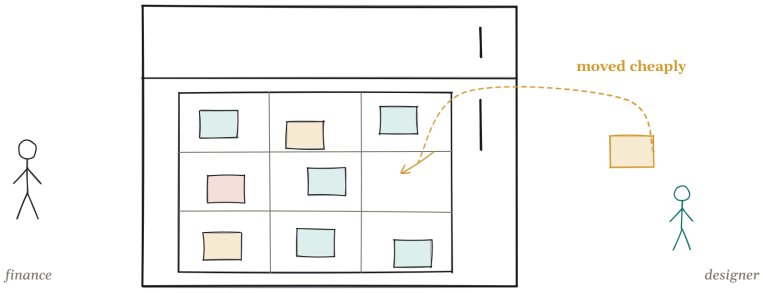


PLATE 8 2000s: The Business Model on One Page

Alexander Osterwalder began his work on business models in a 2004 doctoral thesis. With Yves Pigneur he turned it into the Business Model Canvas, published in *Business Model Generation* in 2010. The canvas has nine blocks: customer segments, value propositions, channels, customer relationships, revenue streams, key resources, key activities, key partners and cost structure.

Its success had little to do with the blocks themselves. Most of them were already in strategy textbooks. It succeeded because of three properties. It fit on one page, so the whole model was visible at once. It gave a room of people a shared vocabulary, so a designer and a finance director could argue about the same box. And it worked with sticky notes, so a team could change its mind cheaply and in public.

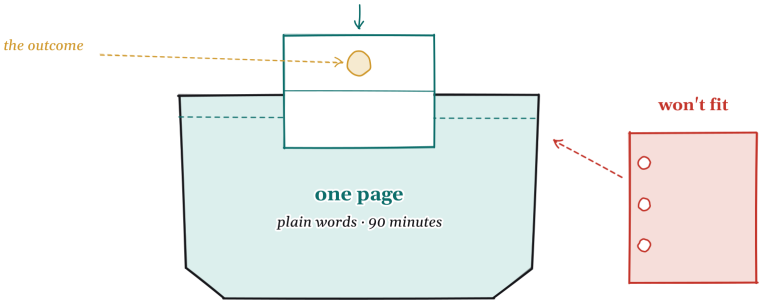


Like a family planner on the fridge door: *the whole week on one sheet, everyone reads the same days, and a magnet moves in a second.*

The canvas arrived when the hard problem in business was finding a model that worked. The web had opened new ways to reach customers and charge them. Founders needed a fast way to sketch, test and compare business models, and the canvas gave them one.

Any successor has to keep those three properties. A new canvas that needs a consultant to read it, or a week to fill in, has already failed. The Work Loop Canvas is built to the same rule. It fits on one page, it uses plain words a process owner already knows, and a team can fill a first draft in ninety minutes.

The Business Model Canvas also left behind the outcome node. Its centre is the value proposition, a statement of what a customer gets. The Work Loop Canvas keeps that habit and moves it inside. Its centre is the outcome, a statement of what one accepted result looks like.



Like a jacket pocket: *a successor canvas must still slip in as one folded page in plain words, while a thick consultant's binder will not fit.*

VI

The Bots

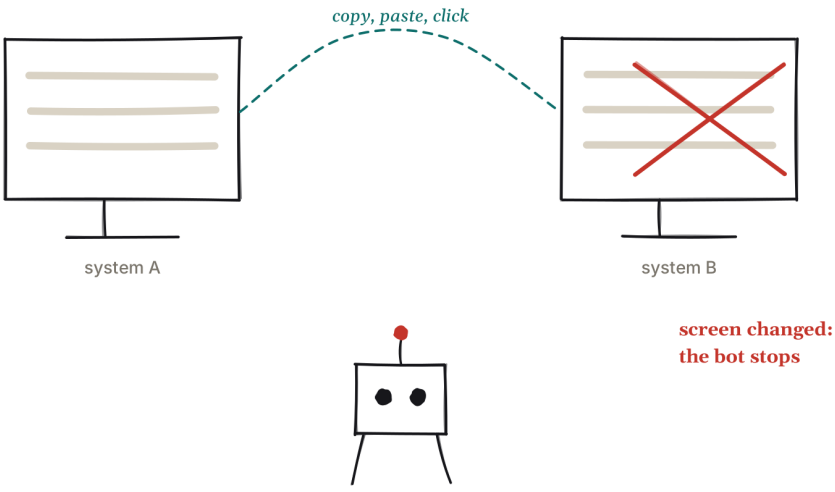
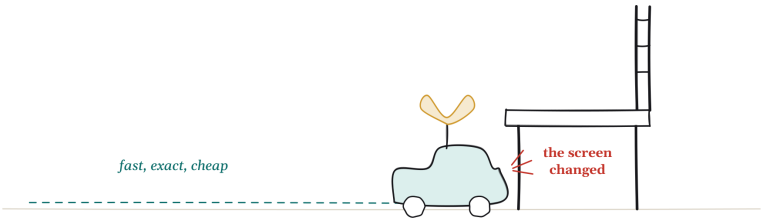


PLATE 9 2010s: The Bots

The 2010s brought robotic process automation. Software robots clicked through the same screens people used, copying data from one system to another and filling in forms. They were fast, cheap to start and needed no change to the systems underneath. A team could automate a dull task in weeks without waiting for the IT department to rebuild anything.

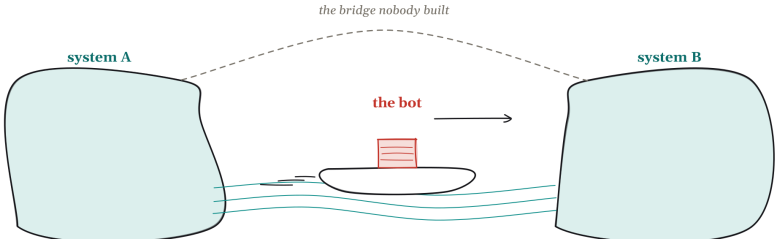
They were also brittle. A bot followed its script exactly. When a screen changed or a case fell outside the script, it stopped. Large programs ended up with teams whose job was to repair bots after every system update.



Like a wind-up toy car: *it races its track fast and exactly, but when someone moves a chair it hits the leg and stops.*

There is a deeper problem. Robotic process automation automated the workaround in the most literal sense. Re-keying data from one system into another is a compensation: it exists because the two systems were never connected. A bot that does the re-keying faster makes the compensation permanent. It recorded how people coped with disconnected systems and repeated it.

The bots left behind one useful part of the canvas, the actions node. They proved that software could act inside business systems: read a record, write a field, trigger a payment. The question they never answered was when it should. A bot has no judgment, so it either acts on everything in its script or nothing outside it.



Like a faster ferry between two banks that were never bridged: *the crossing speeds up, and the missing bridge becomes a permanent habit.*

VII

From Business Model to Business Engineering

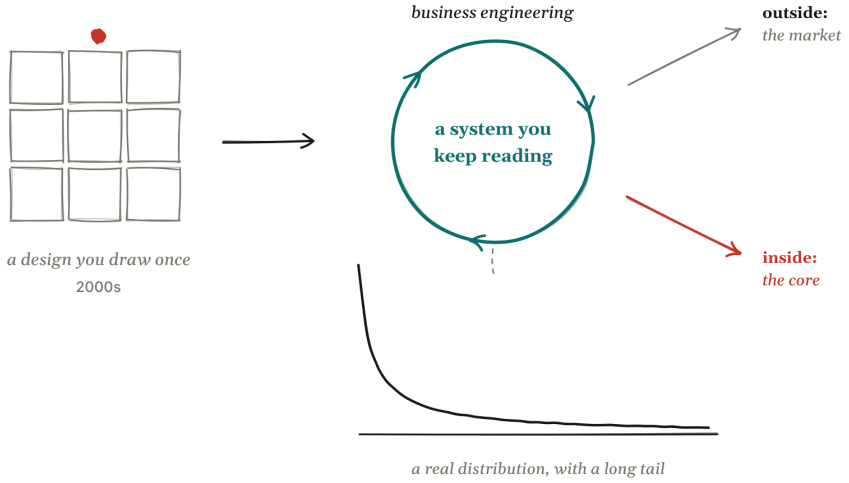
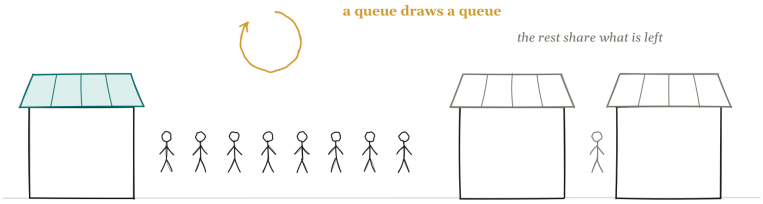


PLATE 10 From Business Model to Business Engineering

The business model canvas treated a business as a design. A team drew it once, tested it with customers and adjusted it. That fit a world where most of a business model was a set of choices facing the market: which customers, which offer, which channel, which price.

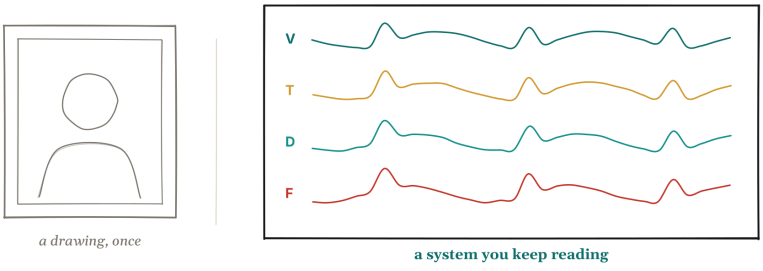
The web and its platforms broke that assumption. In the businesses that grew fastest after 2010, technology and distribution were not supporting blocks. They were the model. A search engine's results improved with every query it served. A marketplace grew more useful with every seller who joined. A software product spread through the people already using it. Growth compounded through loops, and outcomes followed power laws. A few products, customers and companies took most of the value, and the rest shared what was left.



Like stalls at a street market: *people join the longest queue, so it grows longer still, and a few stalls end up with most of the customers.*

A business like that is not a drawing on a page. It is a system running inside a real distribution of customers, cases and outcomes, and it changes every time it runs. The canvas could describe where such a business stood on a given day. It could not say how the business would behave next quarter, because that depended on feedback loops and on the shape of the distribution underneath. Drawing it was not enough. It had to be read for its structure, tested against what actually happened and adjusted as it went.

Business engineering was built to do that. It starts from the real distribution a business operates in, rather than from a list of boxes. It abstracts and compresses what it finds into mental models, tests those models against real cases and builds the intuition to notice when the distribution shifts. Its best-known instrument, VTDF, reads a business through its value, technology, distribution and financial models together, because in a platform business those four move together. **Business engineering turned the business model from a drawing into a system you keep reading.**



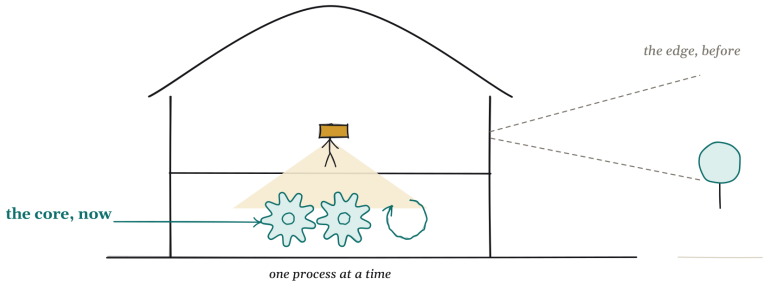
Like a bedside monitor instead of a portrait: *business engineering keeps reading four traces, value, technology, distribution and finance, as they move together.*

That shift is why business engineering is the key transition in this history, and not one more framework in the list. It changed the question from "what is the design?" to "how does this system behave in its real distribution?" Every

earlier school, from Taylor to the business model canvas, answered the first kind of question.

It also matters for a second reason. The habits that read a dynamic business from the outside are the same ones needed to read it from the inside. For the platform era, the dynamic part of a business sat at its edge, in distribution and product. The inside could stay a set of fixed procedures. Once the executor decides, that stops being true. The core of the firm begins to behave the way the platforms did: it runs on loops, it learns, it is uneven, and its tails carry the risk.

A canvas for the core therefore has to come out of business engineering, not out of the static template that came before it. The Work Loop Canvas is business engineering turned inward, applied to one process at a time. Part Two sets out the method in full.



Like a torch swung from the garden to the cellar: *the same light that read the business's edge now reads its core, one process at a time.*

VIII

What Each Era Left Behind

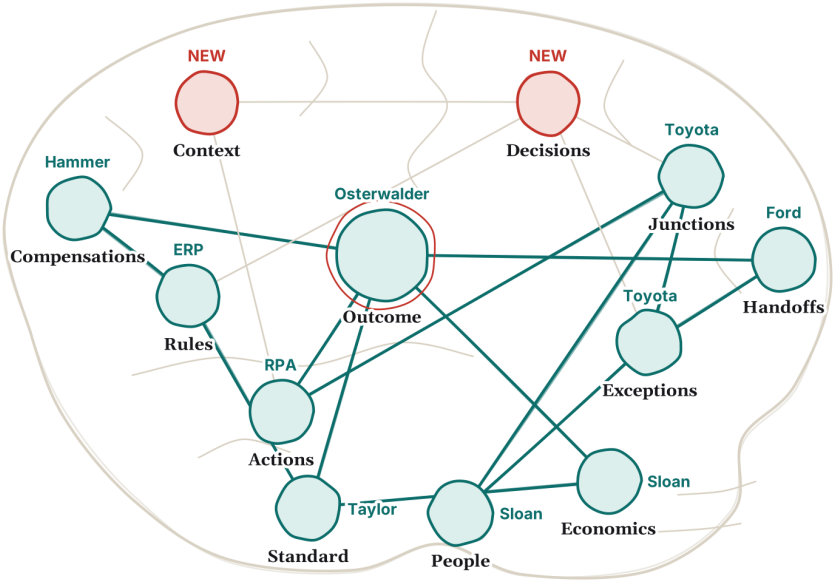
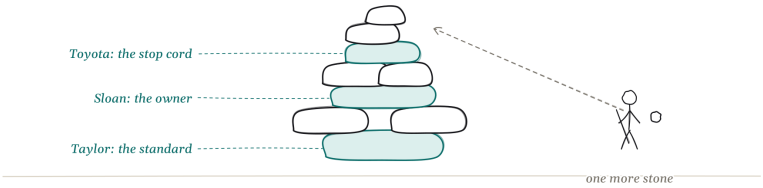


PLATE 11 What Each Era Left Behind

Lay the century side by side and a pattern appears. Each era left behind one of the parts a modern loop needs.

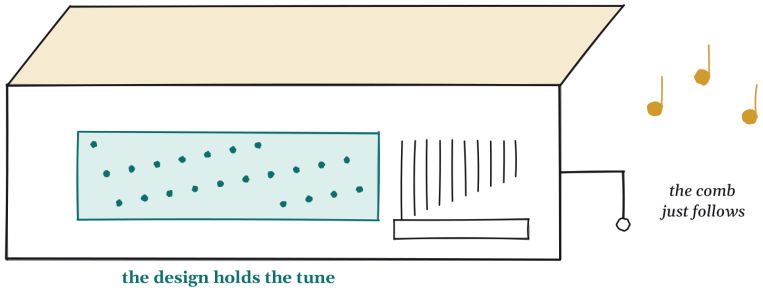
Taylor left the written standard. Ford left the designed flow and its handoffs. Sloan left the named owner and the measured economics. Toyota left the stop cord, the point where a person decides on an abnormal case, and the queue that catches what the line cannot handle. Reengineering left the instruction to strip out what no longer serves the outcome. Enterprise software left rules written into systems. Robotic process automation left software that acts inside them. The business model canvas left the habit of stating value on one page. Business engineering left the method that made the next step possible: reading a business as a running system in a real distribution, not as a fixed design.



Like a cairn built by passing walkers: *each era set down one stone, standard, flow, owner, stop cord, and the pile now marks the way.*

That accounts for ten of the twelve nodes. The two that remain are context and decisions. They are new because the executor is new.

Every earlier generation designed work for an executor that follows instructions. A worker on Ford's line, a clerk following an ERP screen and a software robot all have this in common. They do what the design tells them. The intelligence lives in the design, and the executor carries it out.



Like a music box: *the tune lives in the pinned drum, and the steel comb plays exactly what the pins tell it, nothing more.*

An AI agent is different. It reads context, interprets a case and makes decisions within the bounds it is given. That changes what the designer has to specify. It is no longer enough to describe the steps. The designer has to say what the executor may read, which decisions it may take alone and where a person must decide.

So the agentic era is the first to design for an executor that decides. Most of the parts are old and well understood. The executor is new, and it forces all the old parts into a single design. That design is the work loop.

PART TWO

The Method Beneath the Canvas

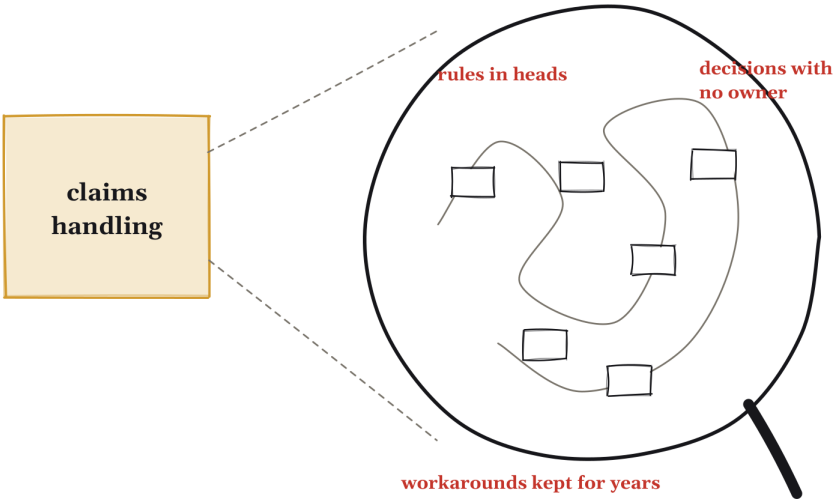
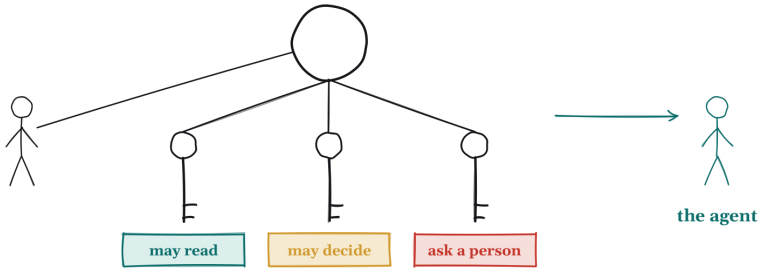


PLATE 12 One Sticky Note Called Key Activities

A canvas is a template, and templates are easy to copy. What makes one useful is the way of thinking behind it. The Work Loop Canvas comes out of business engineering, a discipline for reading how a business actually works. This part sets out that discipline: where it starts, how it moves from observation to model and back, and how its best-known instrument, VTDF, connects to the canvas.



Like handing over a ring of tagged keys: *each tag says what the new holder may open, what they may decide alone, and where to ask.*

IX

Business Engineering, Beneath the Template

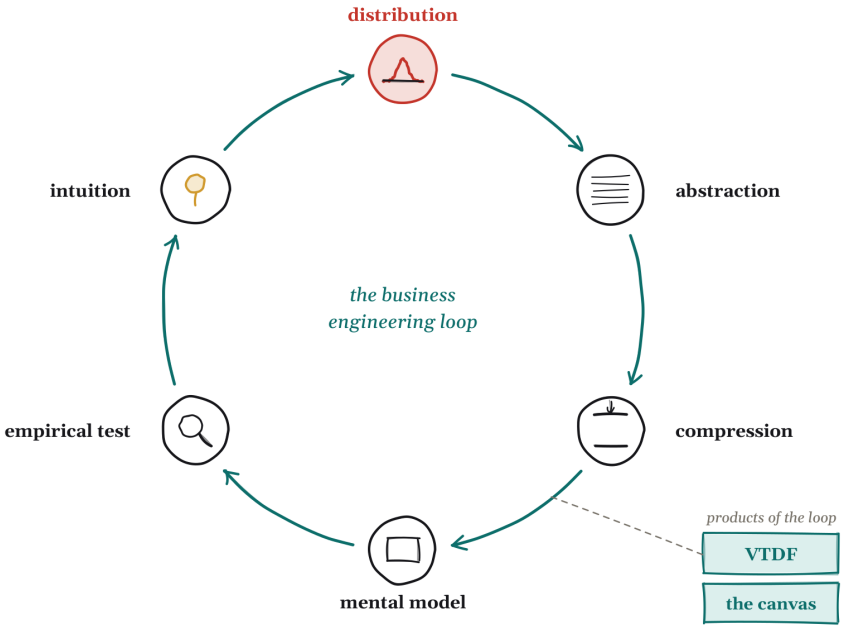


PLATE 13 Business Engineering, Beneath the Template

Most business frameworks are lists of boxes. They tell you what to look at, and they are useful for that. Business engineering starts somewhere else. It starts with the world the business actually operates in, and it treats every framework as a tool for understanding that world, not a substitute for it.

The discipline works as a loop. It begins by understanding the distribution of the domain you operate in: what kinds of cases, customers and outcomes occur, how often, and how large they get. From there it abstracts, stripping away the specifics to see the structure underneath. It compresses that structure into the fewest words that still carry the mechanism. It names the compression as a mental model, so that it can be reused and taught.

Then it goes back to the real world. A mental model is a claim about how something works, and claims have to be tested against cases. Repeated testing

builds intuition, the trained ability to see a situation and know what matters before the analysis is done. And intuition sends you back to the start, because a person with good intuition notices when the distribution has shifted under them.

Distribution, abstraction, compression, mental model, empirical test, intuition, and back to distribution. **Business engineering is that loop, run continuously.** VTDF is one of its products. So is the Work Loop Canvas.

Underneath the loop sit a few habits of mind that the Business Engineer's Thinking OS sets out in full. Structural thinking looks for the mechanism under a story. Contextual precision asks who the analysis is for, why now and how it will be used. Meta-compression extracts the principle that transfers from one case to another. The calibration loop checks every model against what actually happened and adjusts it. The chapters that follow apply these habits to the problem of the core.

The point for this book is simple. A team that copies the canvas without the loop will fill in twelve boxes. A team that runs the loop will notice which boxes matter for its own distribution of cases, and that is where the value is.

X

The Distribution You Operate In

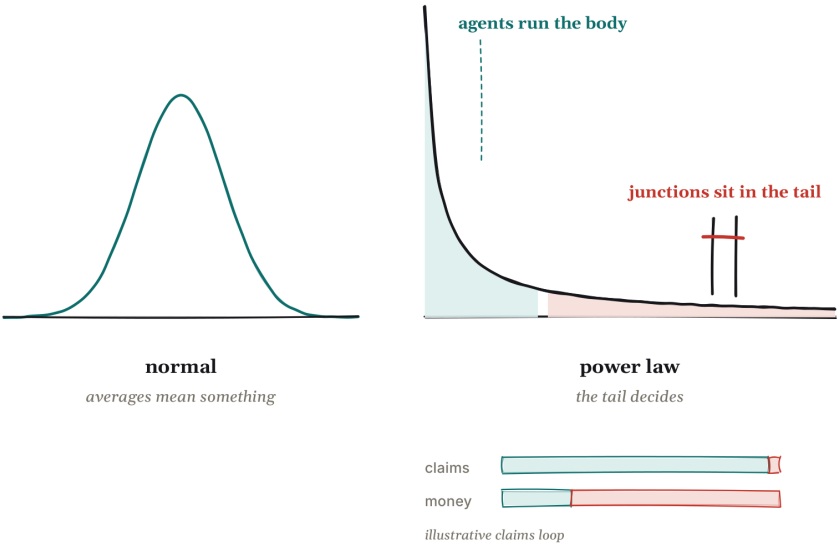


PLATE 14 The Distribution You Operate In

Every business operates inside a distribution, whether or not anyone has drawn it. Here the word means the statistical spread of cases, customers and outcomes, not the Distribution model in VTDF, which is about reaching customers. The most important question about this spread is its shape.

Some domains are close to normal. Most cases cluster around an average, and extreme cases are rare and not much bigger than typical ones. The time it takes to process a standard form, the size of a routine grocery order and the number of documents in a typical claim all behave roughly this way. In these domains, averages mean something. A rule that handles the typical case handles almost every case.

Other domains follow a power law. A small number of cases are far larger than the rest and carry most of the total. Claim amounts, customer revenue, fraud losses, sales of books and films and the value of software projects tend to behave this way. Nassim Nicholas Taleb called the first kind of world Mediocristan and the second Extremistan. In a power-law domain, the average misleads, because it is pulled by cases most people never see.

The shape of the distribution should decide the design of the loop. In the body of a distribution, where cases are many, similar and modest, work can be mechanical or bounded. Rules and thresholds handle it well, and agents can run it alone. In the tail, where cases are few, unusual and expensive, judgment matters more than speed. That is where junctions belong, and where the exception queue earns its keep.

Many failed automation programs made the same mistake. They designed for the average case, measured success on the average case, and were undone by the tail. A loop that settles ninety-five percent of claims correctly can still lose money if the five percent it gets wrong are the large ones.

So before filling a canvas, draw the distribution of the loop's cases. Count them by size, by type and by value. Find where the volume sits and where the money sits, because they are usually in different places. In the insurer's claims loop, about four in five claims were under five thousand dollars, but they carried only about a quarter of the money paid. The largest three percent of claims carried around two-fifths. Those figures are illustrative, and their shape is common. They told the team where to put the agents and where to put the people.

XI

Abstraction, Compression and the Mental Model

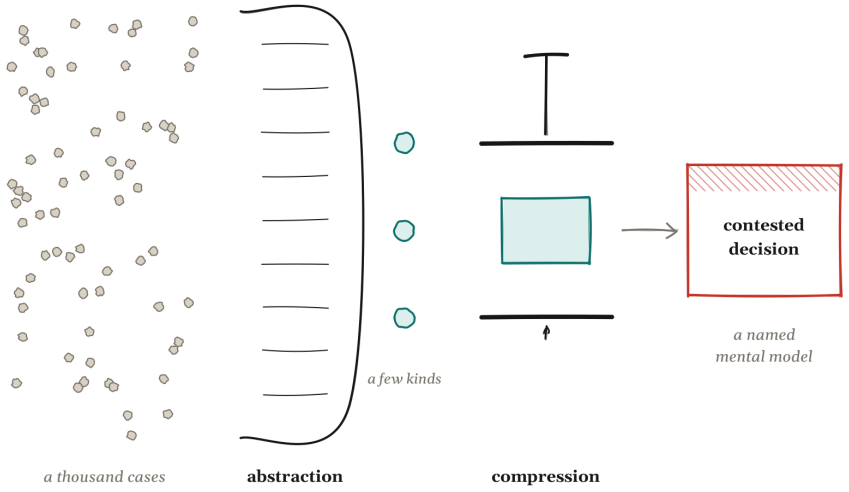


PLATE 15 Abstraction, Compression and the Mental Model

Once the distribution is visible, the next move is abstraction. Abstraction means stepping back from individual cases to see what they share. A thousand claims become a handful of kinds of claim. A hundred process steps become a few kinds of step: those the outcome needs, those that compensate for old constraints and those that make decisions.

Compression goes further. It takes the structure that abstraction revealed and states it in the fewest words that still carry the mechanism. "Most of the time in this process is waiting, not work" is a compression. So is "the tail carries the money." A good compression can be said in a sentence and checked against the evidence. A bad one sounds wise and predicts nothing.

Compression is not summarizing. A summary keeps the topics and drops the detail. A compression keeps the mechanism and drops everything that does not affect it. That is why compressed ideas transfer. The mechanism behind a delayed insurance claim is often the same as the one behind a delayed purchase order or a delayed hospital discharge.

A mental model is a compression with a name. Naming matters because it makes the idea portable. Once a team has the words "contested decision", it can spot one in any meeting, in any department. The Work Loop Canvas is built from mental models of this kind: compensations, the junction, the four-class decision sort, the signed node. Each is a compression of a pattern seen many times, given a name so it can be used.

The canvas itself is a compression. It reduces a process that might fill a hundred-page manual to twelve questions on one page. That is only safe if the twelve questions are the ones the mechanism depends on. The rest of the method exists to check that they are.

XII

Back to the Real World

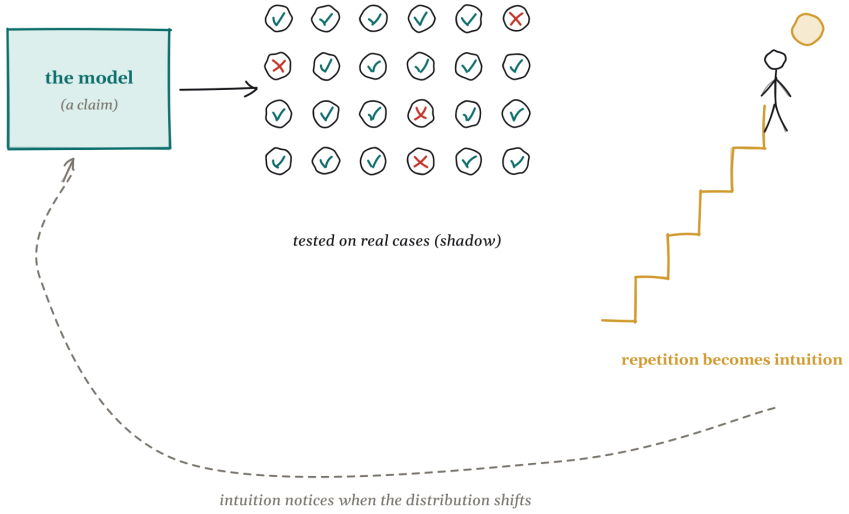


PLATE 16 Back to the Real World

A mental model is a claim, and claims can be wrong. Business engineering treats every model as provisional until it has been tested against real cases, and keeps testing it after that.

The Work Loop Canvas builds the test into its gates. The Shadow gate exists for exactly this reason. A loop in shadow runs beside today's process, makes its decisions on real cases and changes nothing. Every outcome is graded against the standard and compared with what people decided. Where the loop and the people disagree, one of them has a model of the work that is wrong, and finding out which is the most valuable thing the shadow run produces.

Testing also builds intuition. Intuition is not the opposite of analysis. It is analysis that has been repeated so often it no longer needs to be spelled out. The psychologists Daniel Kahneman and Gary Klein, who began on opposite sides of the question, agreed on when intuition can be trusted. It is reliable in environments that are regular enough to learn and that give fast, clear feedback. It is unreliable where outcomes are rare, delayed or dominated by chance.

That finding connects intuition back to the distribution. In the body of a distribution, where cases repeat and feedback is quick, experienced people develop sound intuition, and so can a well-graded loop. In the tail, where cases are rare and each one is different, intuition is weaker, for people and machines alike. That is one more reason to put junctions and careful review where the tail is.

Intuition also closes the loop. People who work a process every day notice early when something changes: a new kind of claim, a supplier raising prices, customers behaving differently. That noticing is the first sign that the distribution has shifted. When it shifts, the abstractions and models built on it need checking again. **The loop never finishes, because the world it describes keeps moving.**

For a work loop, this means the canvas is never complete. The standard reports drift. The exception queue reports new kinds of cases. The weekly review turns repeated exceptions into rules. Each of these is the business engineering loop running at the scale of a single process.

XIII

VTDF: Reading the Business Model

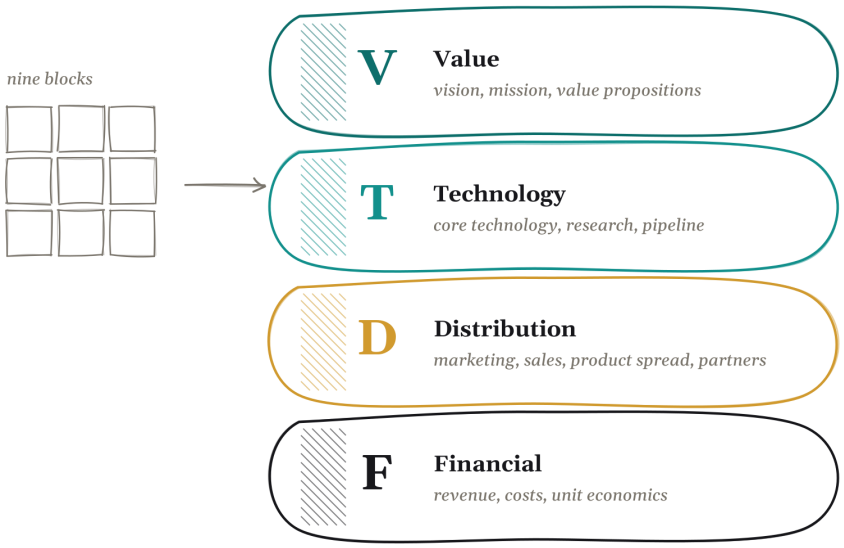
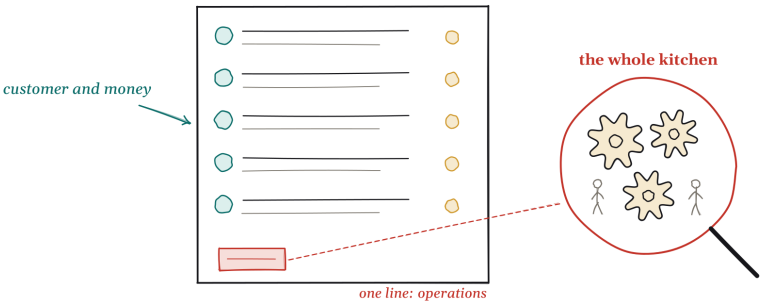


PLATE 17 From Business Model to Business Engineering

The best-known product of the business engineering loop is VTDF, a way of reading a business model through four connected models. It was built because the business model canvas, for all its strengths, left two things in supporting roles that had become central.

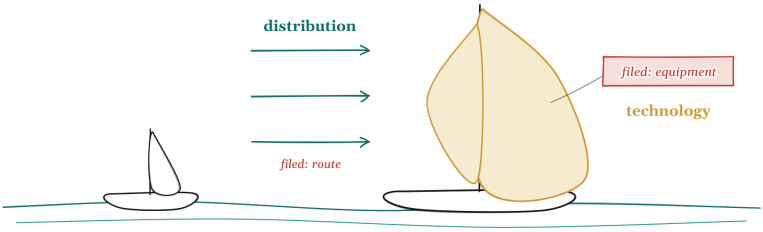
Look at where the Business Model Canvas spends its space. Five of the nine blocks face the customer and the money. Three cover operations: key activities, key resources and key partners. Everything the firm does to produce its product sits in one box, often holding a single sticky note that says "claims handling" or "underwriting".



Like a restaurant menu: *pages on dishes and prices for the diner, while the whole kitchen behind them is squeezed into one small line at the bottom.*

Others saw that gap. In 2017 Andrew Campbell, Mikel Gutierrez and Mark Lancelott published the *Operating Model Canvas*, which unfolds the operational side into value delivery chains, organization, locations, information, suppliers and a management system. It is a good tool for designing how people and systems deliver a strategy. It also assumes, reasonably for its time, that people do the work by following procedures and software does it by following instructions.

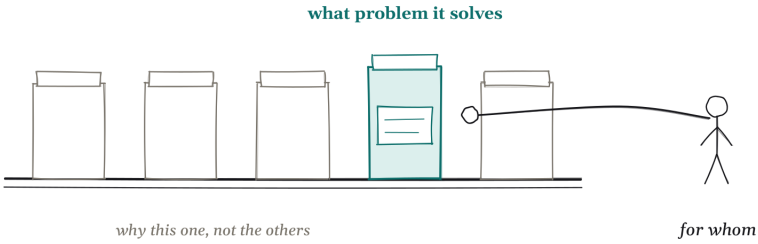
There was a second gap, and it mattered more for the businesses that grew fastest after 2010. Search engines, marketplaces, app stores and software platforms won through product engineering and distribution as much as through their offer. A canvas that treated technology as a key resource and distribution as a channel missed where the advantage sat.



Like a sailing race scored by cargo: *the old form listed the sails as equipment and the wind as a route, yet those two decided who won.*

VTDF reads a business through four models instead.

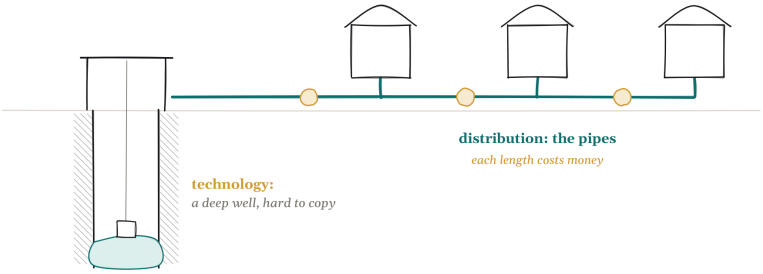
The **Value model** covers the vision, the mission and the value proposition for each kind of customer. It asks what problem the business solves, for whom, and why they would choose it.



Like a shopper reaching past a full shelf for one jar: *the value model says what problem it solves, for whom, and why they pick it.*

The **Technology model** covers the core technology, the research spend and the pipeline of innovation. It asks what the business can build that others cannot easily copy.

The **Distribution model** covers how the business reaches customers: marketing and sales, product features that spread it, partnerships and deals. It asks how the business grows, and what that growth costs.

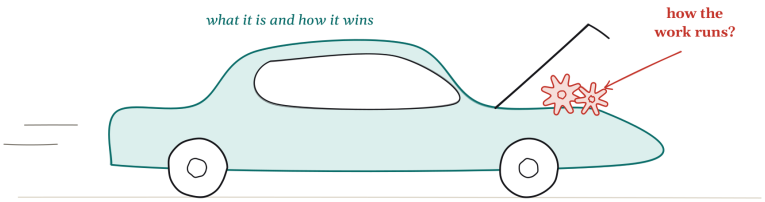


Like a deep well and its pipes: *the well is what rivals cannot easily dig, and the pipes are how water reaches homes, each at a cost.*

The **Financial model** covers revenue streams, cost structure, unit economics and the path to profit. It asks whether the business works as arithmetic.

VTDF puts technology and distribution on the same level as value and money, and it reads the four together, because in a modern business they move together. Like every product of the loop, it is a compression. It keeps what drives a business model and leaves out the rest.

VTDF still describes what the business is and how it wins. It says much less about how the work inside it runs, and almost nothing about how that work changes once agents join. That question needs its own instrument, one level down.



Like a car seen from the kerb: *you can tell what it is, how it wins and what it costs, but how it runs needs the bonnet lifted.*

XIV

Where the Work Loop Plugs In

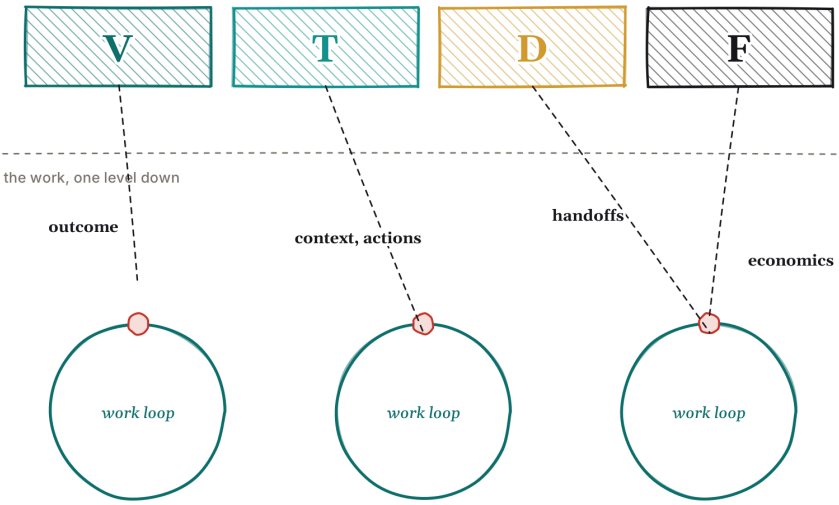
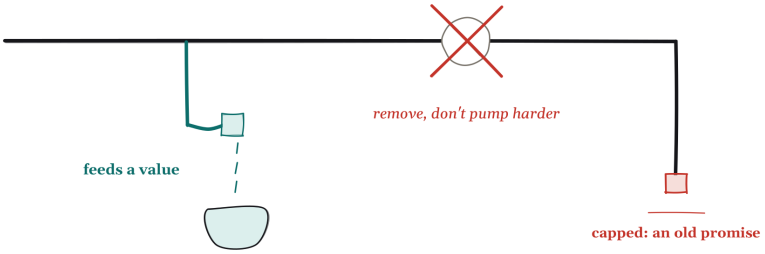


PLATE 18 Where the Work Loop Plugs In

The Work Loop Canvas sits underneath VTDF. The two connect at four points, and each connection is a check a team can run.

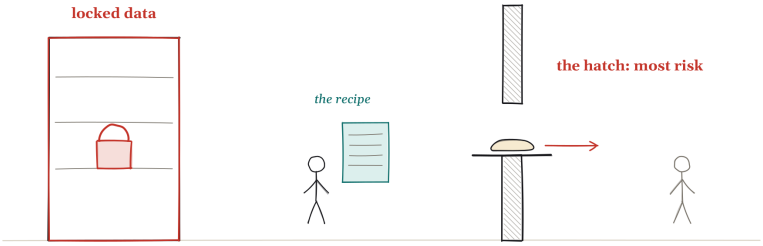
The first connection is value. Each loop's outcome should trace to a value proposition in the Value model. If a loop serves no value proposition, it is a candidate for removal, not automation. Many back-office loops fail this test when it is first applied. They exist to serve other internal loops, which exist to serve a customer promise that has since changed.



Like old plumbing traced tap by tap: *a pipe feeding a working tap stays, and one ending at a capped stub is removed, not given a bigger pump.*

The second connection is technology. The Technology model sets what the firm's executors can read and change. In canvas terms, it fills the context and actions nodes. A loop that needs data the firm cannot reach, or needs to write to a system nobody can safely open, is blocked at the technology level. No amount of process design will fix it.

The third connection is distribution. The Distribution model meets the loop at its handoffs, wherever a loop passes work to a customer, a partner or a channel. These handoffs carry the most risk, because a mistake there reaches someone outside the firm.



Like a kitchen with a locked pantry and a serving hatch: *no recipe helps when the pantry is locked, and the hatch is where a mistake reaches the diner.*

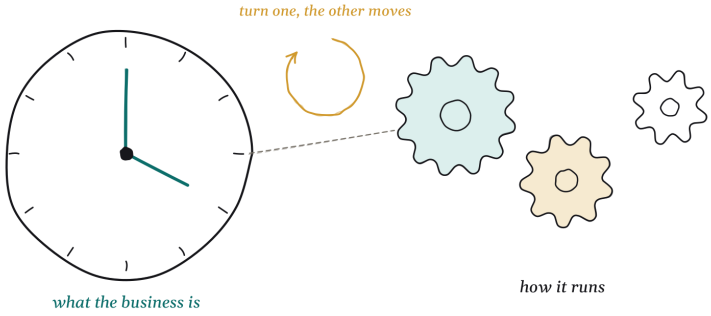
The fourth connection is money. The Financial model receives each loop's economics. Cost per accepted outcome, added up across all the loops, is the firm's cost structure seen from the inside. When the numbers do not add up, either the loops are badly measured or the cost structure includes work no loop accounts for.

So the two instruments work as a pair. **VTDF says what the business is. The work loops say how it runs.** A change in one should show up in the other. A new value proposition should create or change a loop. A loop that gets much cheaper should change the Financial model, and perhaps the price.

PART THREE

The Inside-Out Problem

The method says what to look for. This part says why the core, rather than the market, is where AI now has to work, and why the unit of design has to change.



Like a clock's face and its gears: *the face says what the business is, the gears say how it runs, and turning one moves the other.*

XV

Outside-In, Inside-Out

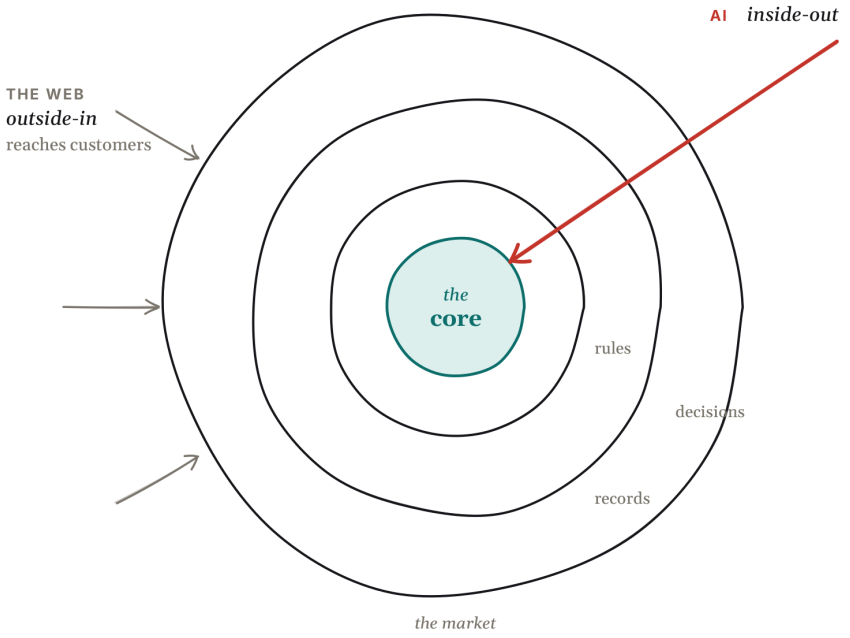
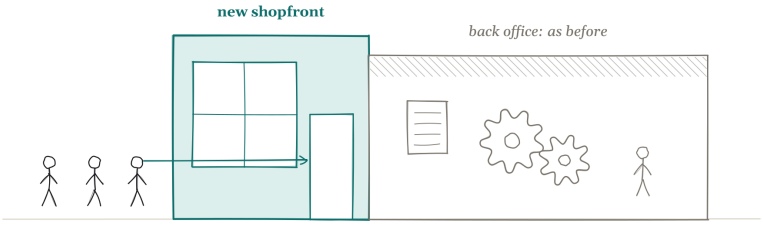


PLATE 19 Outside-In, Inside-Out

The web was an outside-in technology. It changed how a firm met its market: how customers found it, bought from it and talked about it. Internal processes could stay largely as they were. A bank put its products online, and the back office kept running the same way. A retailer opened an online store, and the warehouse still picked orders from a list.

That is why the business model canvas, and later VTDF, focused so much on the outside. The advantage of the web era sat at the edge of the firm, in distribution, discovery and the customer interface. The Business Engineer has long described the relationship between the two technologies as Web Squared. AI does not replace the web. It compounds it, working from the opposite direction.

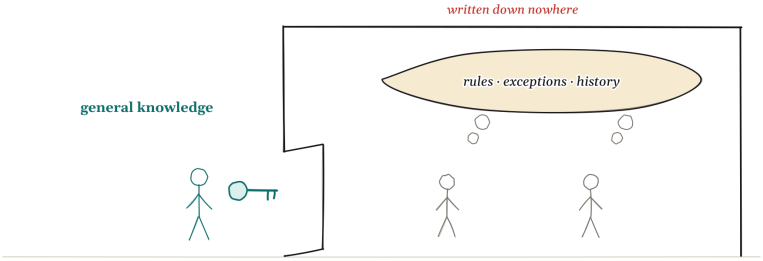


Like a shop given a new glass front: *the web changed how customers found and bought from the firm, while the workshop behind ran exactly as before.*

AI works from the inside out. Its value depends on entering the processes themselves: reading the case file, applying the rules, making the decision and writing to the system of record. A chatbot on the website is the outside-in use of AI, and it is the easy part. The hard part is the claim, the loan, the order and the invoice.

An AI model arrives with broad general knowledge and no knowledge of the firm. It does not know the firm's rules, its exceptions, its customers' history or which decisions need a manager. To be useful, it has to get inside. The inside is exactly where most firms are least documented, because for decades the people doing the work carried that knowledge for them.

So the frontier has moved. In the 2000s the scarce skill was designing the business model. Today it is mapping the core well enough for a new kind of worker to operate in it. **The hard problem moved from the business model to the process.**



Like a gifted new hire at the door on day one: *they know the trade, but the firm's rules and exceptions live only in the heads of the old hands.*

XVI

Why AI Stalls at the Edge of the Core

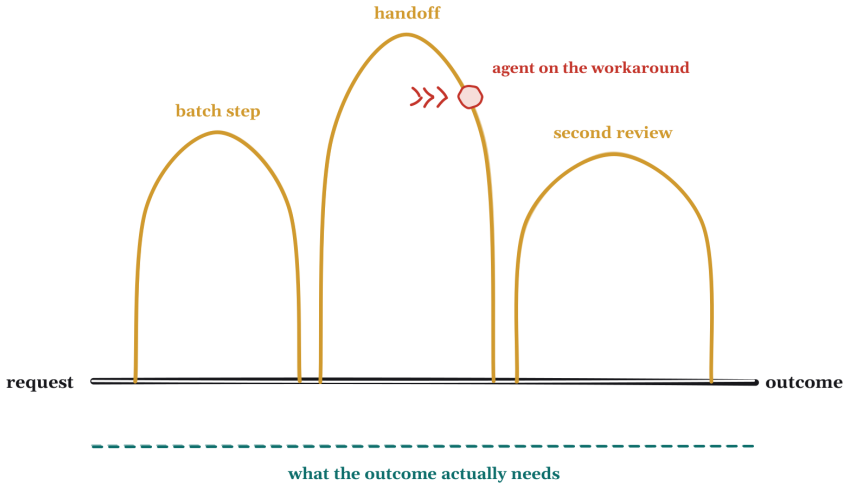
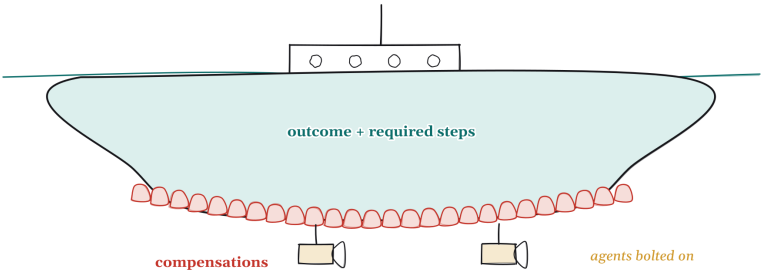


PLATE 20 Why AI Stalls at the Edge of the Core

When a company tries to put AI into a core process, it usually starts from the process map it already has. The map shows a sequence of tasks. The team picks the tasks that look automatable and attaches an agent to each one.

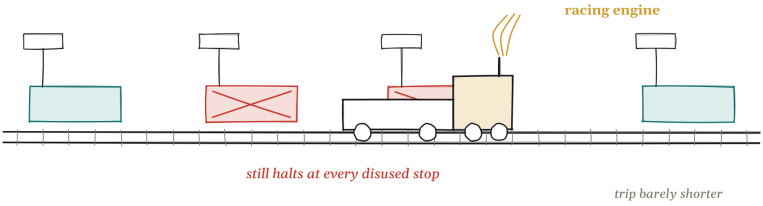
The results disappoint for a specific reason. A documented process is three things fused together. There is the outcome the process exists to produce. There is the minimum set of steps that outcome really requires. And there is a thick layer of compensations: steps that exist only to work around constraints that no longer apply.



Like a hull thick with barnacles: *the outcome and its real steps lie under years of workarounds, and the team bolts its new motors onto the crust.*

A batch step exists because a system once ran only at night. A handoff exists because two teams once sat in different buildings. A second review exists because an earlier step used to be unreliable. A form exists because a person once had to be prompted to collect the right information. Years of optimization polish these steps until they look structural. They appear on every map, they have owners and metrics, and nobody remembers why they started.

Attach an agent to them and it runs the workaround at machine speed. The pilot shows a gain on one task. The process as a whole barely moves, because most of its time and cost lived in the parts nobody questioned. This is the same mistake Hammer described in 1990, made with a more powerful tool.

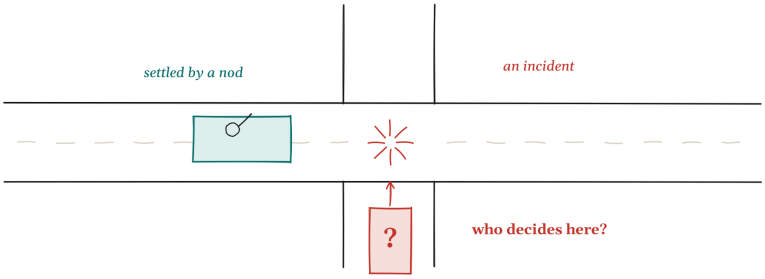


Like a racing engine in a train that still halts at every disused station: *one leg gets faster, but the journey barely shortens.*

There is a second reason pilots stall, and it matters more. Agents do not only follow steps. They make decisions. A process map shows who does each task. It almost never shows who owns each decision, which decisions a machine may take alone, and which ones the organization has never actually agreed on. Those questions stay invisible while experienced people resolve them quietly. They surface the moment something else has to act on them, usually as an incident.

The third reason is ownership. A pilot belongs to an innovation team or a technology team. The process belongs to an operations leader. When the pilot

needs a rule changed, a junction staffed or a budget moved, it has to cross that line, and it usually stops there.



Like an unwritten right of way at a country crossroads: *locals settle it with a nod, but a driverless car arriving there finds no rule at all.*

XVII

The Unit Is the Work Loop

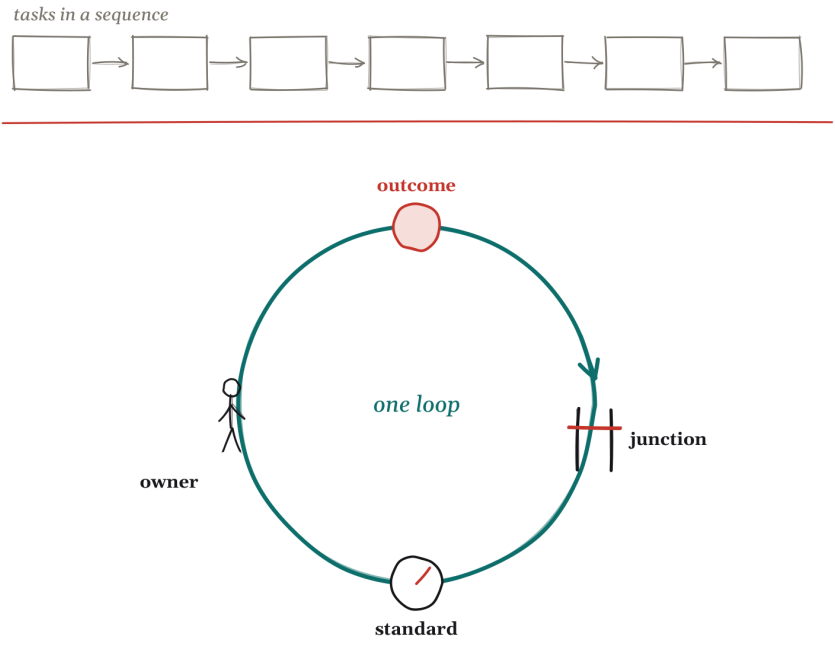
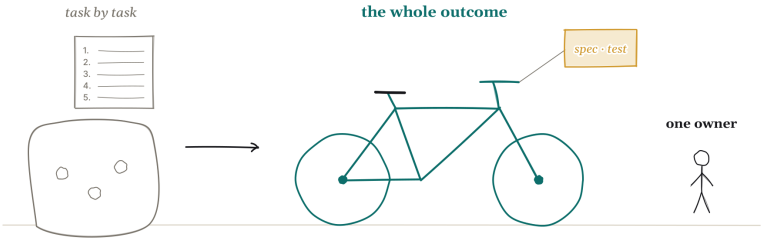


PLATE 21 The Unit Is the Work Loop

The fix starts with changing what gets mapped. The task was the right unit for an executor that follows instructions, because instructions are written task by task. For an executor that decides, the right unit is the **work loop**.

A work loop is an outcome together with everything needed to produce it well. That includes the rules and context it depends on, the decisions it contains and the points where a person must decide. It also includes the standard every result is graded against and the person who answers for all of it.

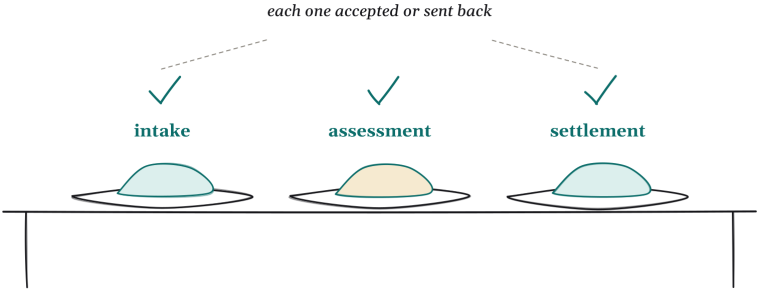


Like ordering a finished bicycle instead of a bag of parts: *the unit is the working bike, with its spec, its test ride and one maker answering for it.*

A work loop is smaller than a department and larger than a task. A claims operation might contain three: intake, assessment and settlement. Each produces something a customer or colleague can accept or reject. Each can be handed to a mixed team of people and agents as a whole, with one owner.

A useful test for the boundary of a loop is the acceptance test. If you can say what an accepted result looks like, and someone downstream can check it, you have a loop. If the only way to judge the result is to look at what happens three steps later, the loop is drawn too small. If the loop produces several different kinds of result for different customers, it is drawn too large.

Redesigning around loops tends to produce a few loops running in parallel, not a swarm of agents. That is easier to govern and easier to measure. It is also easier to explain to the people whose work changes, because each of them can see which loop they belong to and what it is for.



Like the courses of a dinner: *each plate is a whole thing a guest can accept or send back, larger than a garnish, smaller than the banquet.*

XVIII

Shaped Like the Organization's Brain

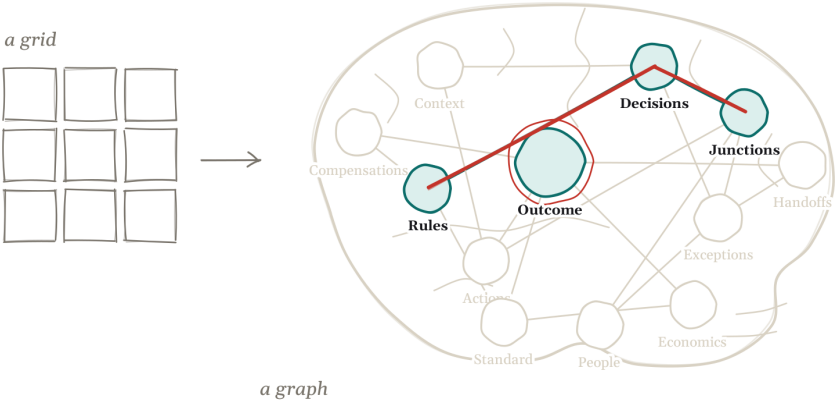
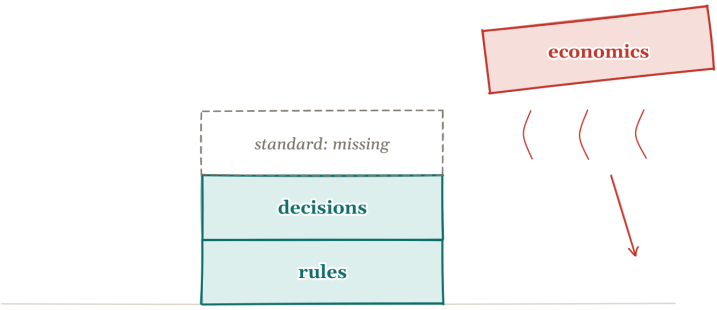


PLATE 22 Shaped Like the Organization's Brain

The Business Model Canvas is a grid, and a grid suits its subject. A business model is a set of choices that sit side by side. A team can fill the blocks in almost any order.

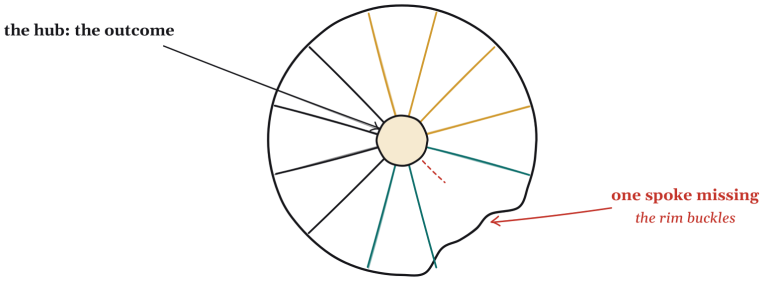
A work loop is different, because its parts depend on each other. A decision cannot be sorted until its rules are known. A junction cannot be placed until the decision behind it is sorted. The economics mean nothing without a standard that says what counts as an accepted result. Drawn as a grid, those dependencies disappear, and teams fill the easy blocks first.



Like stacking building blocks: *each rests on the one below, so a team that places the easy top block first leaves it hanging in the air.*

So the Work Loop Canvas is drawn as a graph. Twelve nodes sit in three regions around a single outcome, with lines between the nodes that depend on each other. One region holds what the loop knows and can touch. Another region decides. A third keeps the loop honest once it runs. The outcome sits in the middle, where every line leads.

The shape resembles a brain on purpose. In a brain, capability comes from regions wired together, not from any single region. The same is true of a loop. A strong rules node is useless if the decisions node is empty. A good standard is useless if nobody owns the loop.



Like a bicycle wheel: *spokes in groups around one hub, and a strong spoke is no use when its neighbour is missing, because the rim buckles there.*

The resemblance also does useful work in a room. When a node is signed, its connections light up. When a line stays dark, everyone can see which dependency is missing. Seen across the whole firm, the canvases form a network of small brains joined by handoffs. That is a fair description of how an organization actually thinks.

PART FOUR

The Canvas, Node by Node

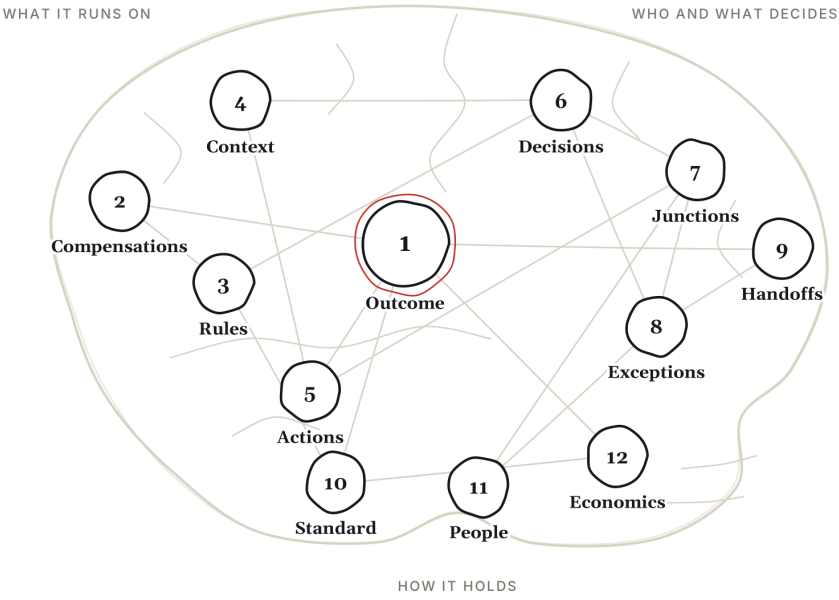
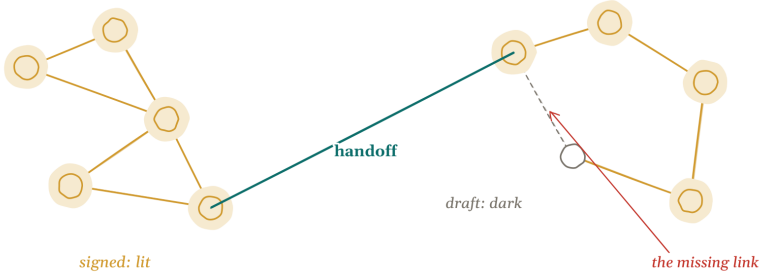


PLATE 23 The Twelve Nodes

This part walks through the twelve nodes in the order a team should fill them. The order matters, because later answers depend on earlier ones. Each chapter gives the node's question, how to fill it, the test for when it is signed and the way it usually goes wrong. The claims assessment loop of the composite insurer runs through every chapter.

A node counts as **signed** only when its answer is written down, agreed and owned by a named person. Until then it is a draft. Draft is a sentence. Signed is a name next to it.



Like lanterns strung across a harbour: *each signed node lights its lines, a dark stretch shows the missing link, and handoffs join one cluster to the next.*

XIX

Outcome

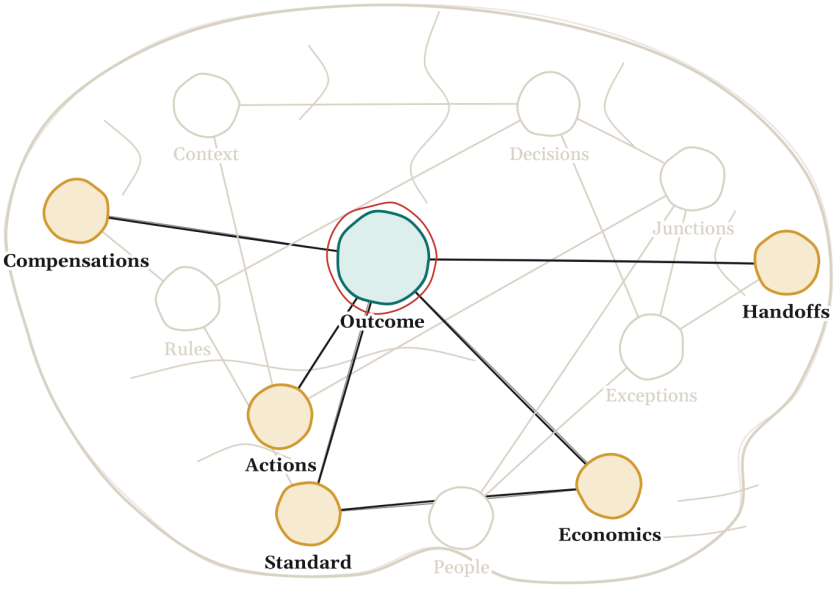
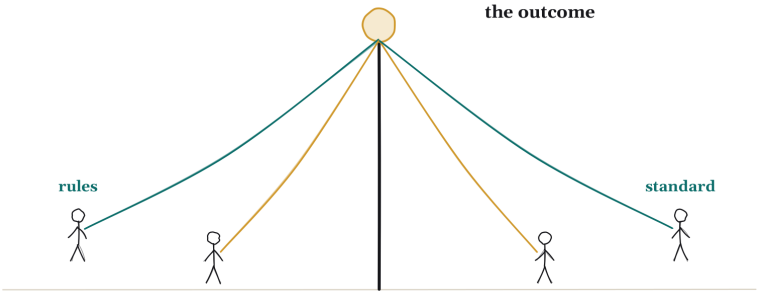


PLATE 24 Outcome

What does one accepted result look like, and who receives it?

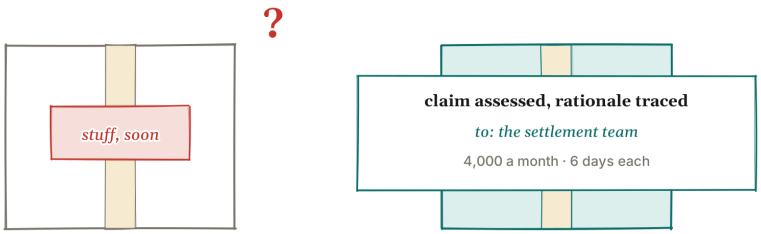
The outcome sits at the centre of the graph because every other node exists to serve it. It is the one sentence that defines the loop. Without it, rules have nothing to serve, the standard has nothing to grade and the economics have nothing to count.



Like a maypole: *every ribbon is tied to the one pole in the middle, and without the pole the ribbons have nothing to hold and fall in a heap.*

Write the outcome as a sentence a customer or colleague would recognize. "Process claims efficiently" is not an outcome. "A claim assessed, with liability and amount decided and a rationale that traces to the rules, ready for settlement" is one. The difference is that someone downstream can check the second one.

Add three facts beside the sentence: how many outcomes the loop produces in a month, how long one takes today and who receives it. The receiver matters more than it seems. A loop whose receiver is another internal team behaves differently from one whose receiver is a customer, because the cost of an error lands in a different place.

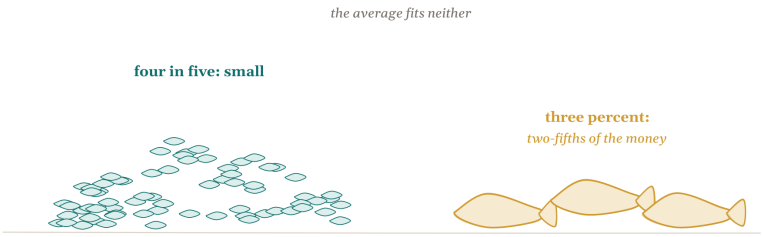


Like a parcel label: *'stuff, soon' gets lost on the way, while named contents and a named recipient let anyone downstream check it arrived right.*

Then draw the distribution of the cases the loop handles. Count them by size, type and value, and mark where the volume sits and where the money sits. This one picture shapes nearly every later node. It shows which decisions can run alone and where people are needed, and it shows which average will mislead the economics.

The outcome is signed when someone outside the team can tell an accepted result from a rejected one. The usual failure is an outcome written as an activity ("assess claims") rather than a result. Activities cannot be graded, so the standard node will stall later.

In the claims loop, the outcome sentence above was agreed in the first fifteen minutes. The volume was about four thousand claims a month. About four in five were under five thousand dollars, and the largest three percent carried around two-fifths of the money paid. Assessment took six of the claim's eleven days. The receiver was the settlement team, and through them the customer.



Like a catch sorted on deck: *a heap of small fish makes up most of the count, while a few big ones carry much of the value.*

XX

Compensations

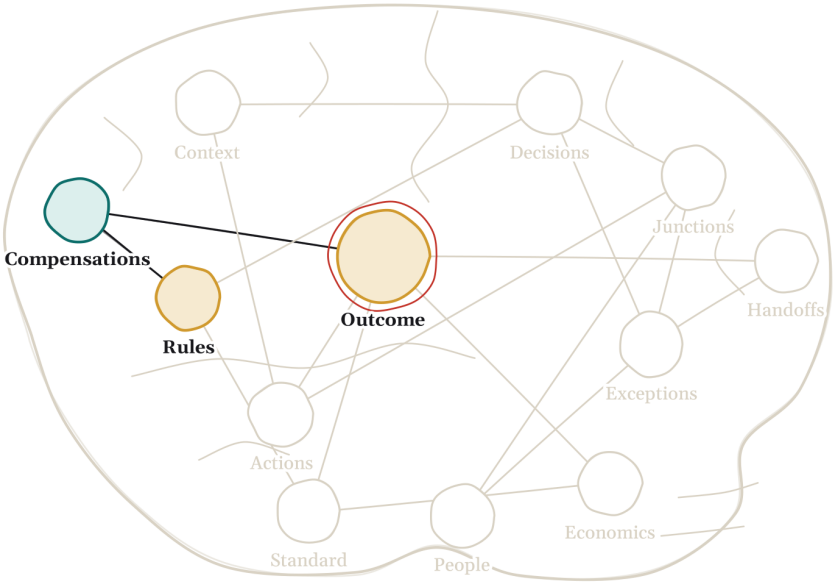
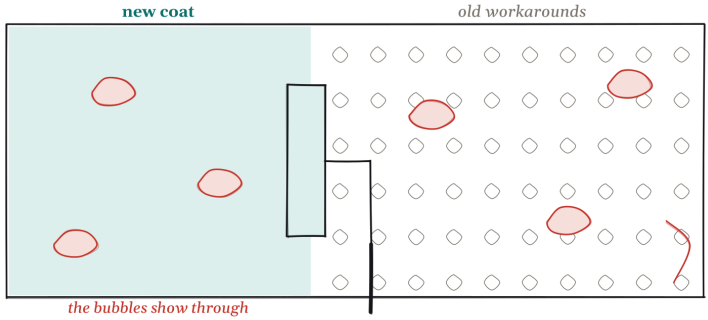


PLATE 25 Compensations

Which steps exist only to work around a constraint that no longer applies?

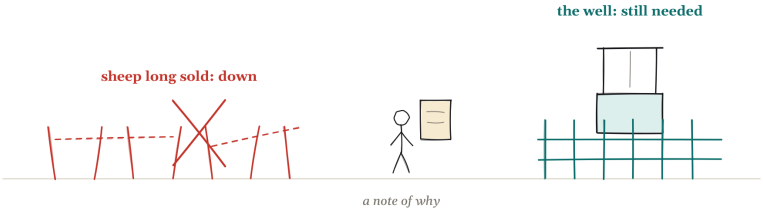
This node comes second for a reason. If a team writes rules and places agents before it strips the compensations, it will carry every workaround into the new design. Reengineering made this argument in 1990. The canvas makes it a required step.



Like painting over old wallpaper: *skip the stripping and every bubble and tear of the old workarounds comes straight through the new coat.*

Walk the current process from end to end and ask of every step: what constraint created this? Then ask whether that constraint still binds. Typical answers include a system that once ran only at night and teams that once sat apart. Others are a form that once prompted a person to collect information, a queue that once rationed scarce attention and a review that once caught errors from an unreliable step.

Strike the steps whose constraints are gone. Keep a short note of each one and the reason, because someone will ask later why it disappeared. Keep the steps the outcome genuinely needs, even when they are slow. A site inspection for structural damage is slow, but the outcome requires it.

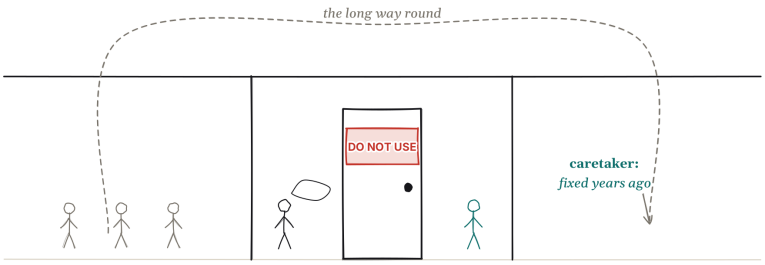


Like walking a farm and asking why each fence was built: *the pen for sheep long sold comes down, while the fence round the well stays.*

The node is signed when every remaining step is required by the outcome itself. The usual failure is deference. Teams hesitate to strike a review that a senior person introduced, or a step a regulator once asked for. The fix is to check: ask the senior person, read the regulation. Most compensations survive only because nobody asks.

In the claims loop, three steps went. The overnight re-score was struck because the rating system now runs on demand. The second review of every claim

under two thousand dollars was struck because the re-keying errors it caught no longer happen. The paper file transfer between intake and assessment was struck because both teams use the same system.



Like a 'do not use' sign on a door fixed years ago: *everyone walks the long way round until someone simply asks the caretaker.*

XXI

Rules

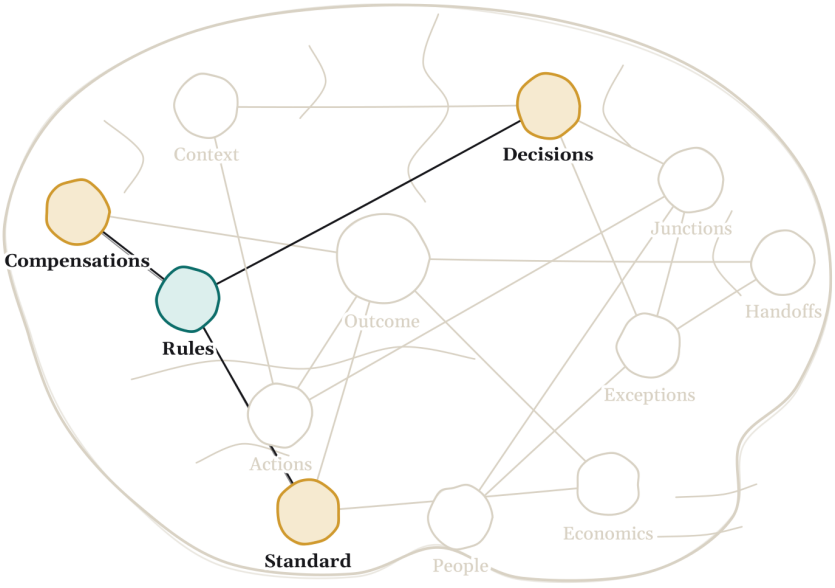
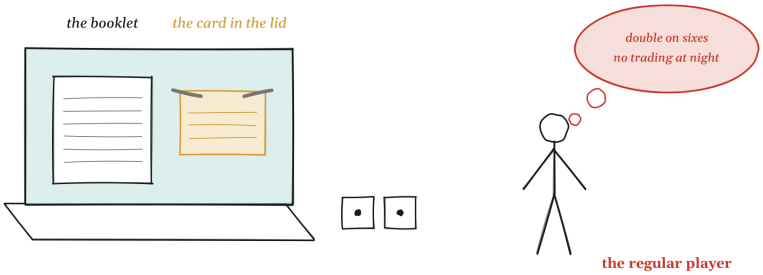


PLATE 26 Rules

Where does each rule live today, and who owns it?

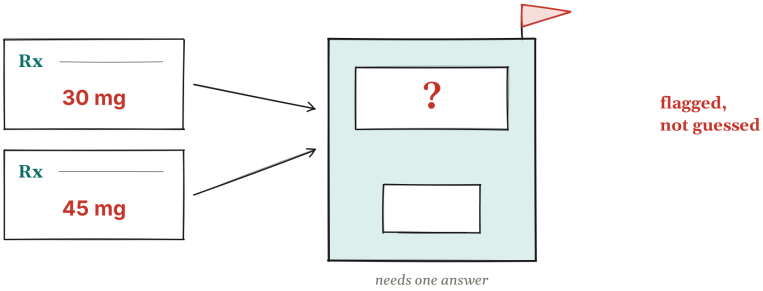
Rules are the instructions a decision has to follow: coverage conditions, approval limits, eligibility criteria, routing logic. In most firms they live in three places at once. Some are in policy documents. Some are in system configuration, written there during an enterprise software project years ago. Many live only in experienced people's heads.



Like a family board game: *some rules sit in the booklet, some on a card taped in the lid, and many only in the regular player's head.*

Gather them in one list. Interview the people who do the work as well as the people who wrote the policies, because the two groups often describe different rules. Read the configuration tables with someone who knows them. For each rule, record where it lives, who owns it and when it was last confirmed.

Contradictions will appear. Treat them as findings, not noise. A contradiction between the manual and the system means that, for some cases, the firm has been applying two different rules without knowing it. An agent cannot do that quietly. It needs one answer.

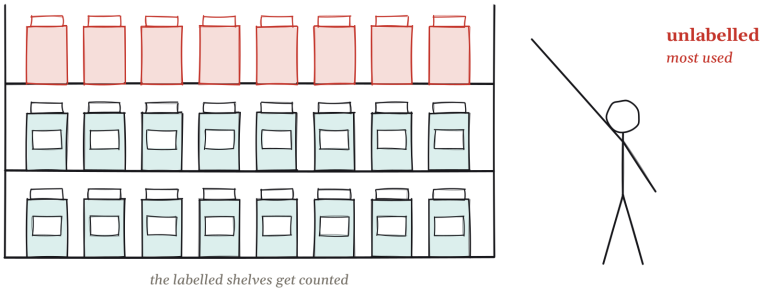


Like a pharmacy given two prescriptions with different doses: *a person might quietly pick one, but the dispensing machine needs one answer, so the clash is flagged.*

The node is signed when every rule has an owner and a date, and the contradictions are listed with an owner for each. The usual failure is stopping at the documents. The rules that matter most are often the unwritten ones, the judgment calls experienced staff make so routinely that they no longer notice them.

In the claims loop, the team found 38 rules. Twenty-one were in the claims manual, nine in the rating engine's settings and eight only in senior adjusters'

heads. Seven contradicted each other. In one, the manual covered water damage after a property had been empty for thirty days, while the engine used forty-five. The Head of Claims Policy took ownership of the list.



Like stocktaking a pantry: *counting only the labelled jars misses the unlabelled ones on the top shelf, which are the ones the cook reaches for most.*

XXII

Context

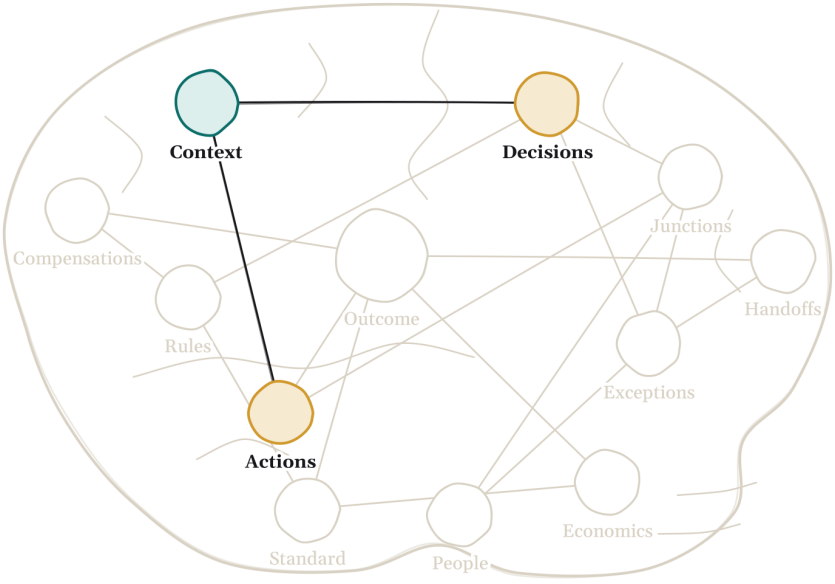
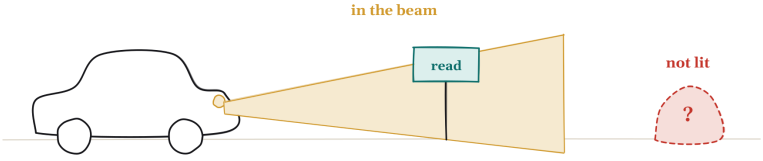


PLATE 27 Context

What must the executor read to act well?

Context is new to the canvas. No earlier generation needed it, because the executor either knew the context already, as an experienced worker does, or did not need it, as a script does not. An agent needs it spelled out. It can only act on what it can read.

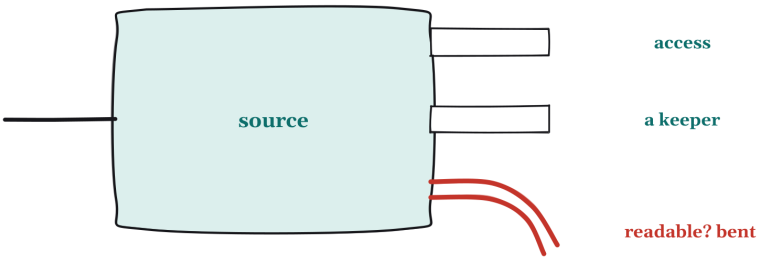


Like driving at night: *the car can only respond to what its headlights reach, so whatever the agent needs must be lit up for it to read.*

List every source a skilled person consults to do the work: records, documents, history, reference data, notes from colleagues. Then ask, for each one, three

questions. Can the executor reach it with the right permissions? Who keeps it current? Is it in a form a machine can read, or is it a scanned page, a photo or a spreadsheet on someone's laptop?

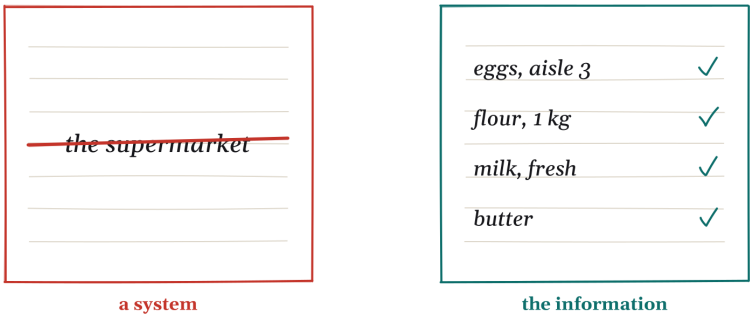
Gaps here are common and usually fixable. They are also the most frequent reason a pilot that worked on clean test data fails on real cases. The real cases depend on a source nobody listed.



Like a three-pin plug: *every source needs access, a keeper and a machine-readable form, and one bent pin stops the appliance working on real cases.*

The node is signed when each source has a keeper and the executor can reach it with the right permissions. The usual failure is listing systems instead of information. "The claims system" is a system. "The prior claims on this policy, from the claims system" is information an assessment actually uses.

In the claims loop, the sources were the policy and its endorsements, the claim notes and photos, repair estimates, prior claims on the policy and a supplier price list kept by procurement and refreshed monthly. Inspection reports arrived as scanned documents. Operations agreed to convert them on arrival, and the node was signed.



Like a shopping list: *'the supermarket' names a place, while 'eggs, flour, milk' names what the recipe actually uses, and only the second list gets you cooking.*

XXIII

Actions

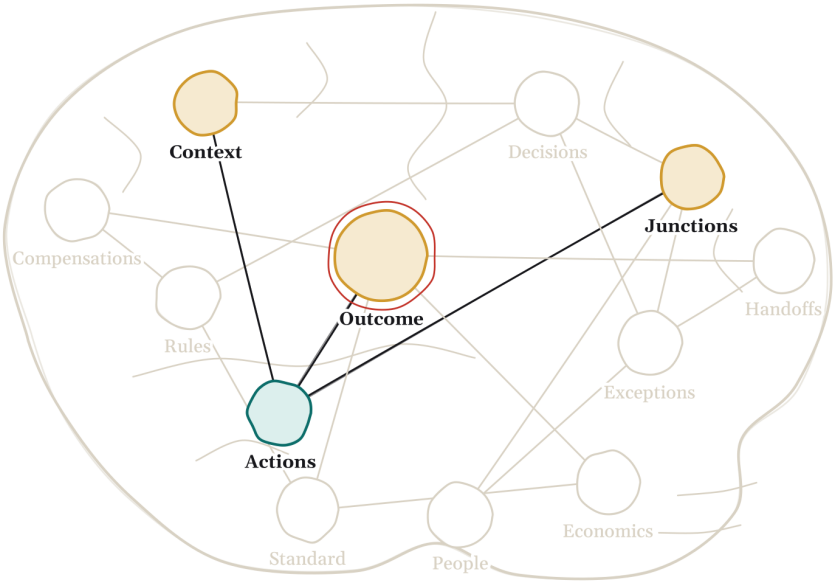
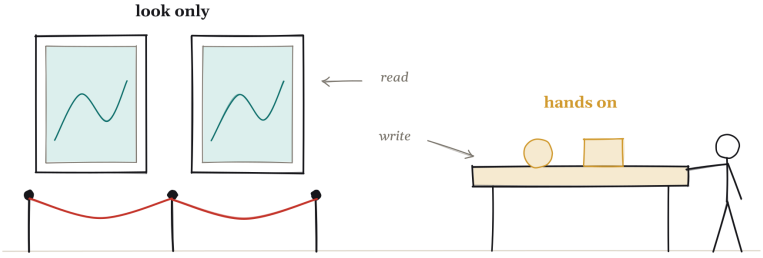


PLATE 28 Actions

What can the executor change, and in which systems?

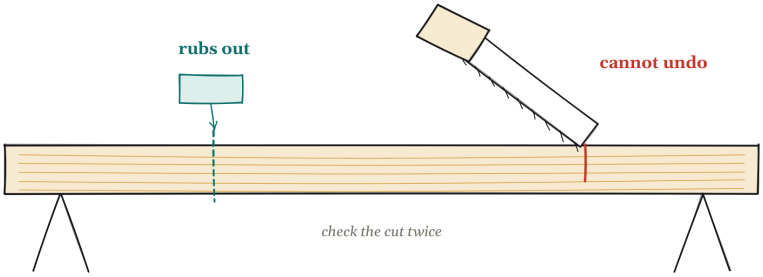
Robotic process automation proved that software could act inside business systems. This node decides what this loop's executor may actually do. It separates reading from writing, system by system.



Like a museum visit: *you may look at every painting, but you may touch only the hands-on table, and the rules differ room by room.*

List every system the loop touches. For each one, record whether the executor reads from it, writes to it or both. For each write, record the permission it needs and whether the action can be undone. Undoing matters most. A reserve figure can be changed tomorrow. A confirmed supplier booking or a released payment often cannot.

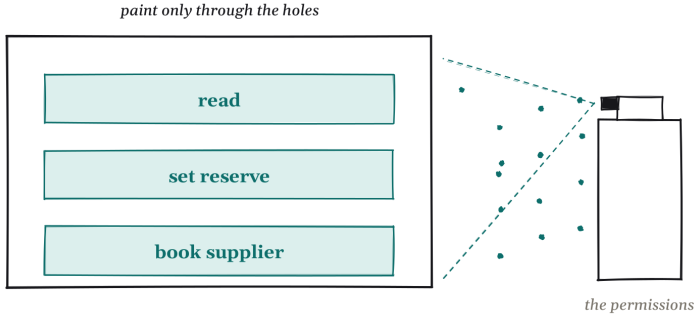
This list feeds the junctions node later. Every action that cannot be undone, or that costs a lot when wrong, is a candidate for a junction.



Like measure twice, cut once: *a pencil mark on the plank rubs out but the saw cut does not, so the cut earns the extra check.*

The node is signed when every write is listed with its system, its permission and whether it can be undone. The usual failure is granting broad permissions for convenience during a pilot and never narrowing them. A loop in production should be able to do exactly what its canvas says and nothing more.

In the claims loop, the executor reads from policy administration, the claims system and the document store. It writes to the claims system, where it sets the reserve and records the assessment, and to the supplier booking system. Reserve changes can be undone. A confirmed supplier booking cannot.



Like a stencil: *paint lands only where the shape is cut out, so the loop does exactly what its canvas says and nothing more.*

XXIV

Decisions

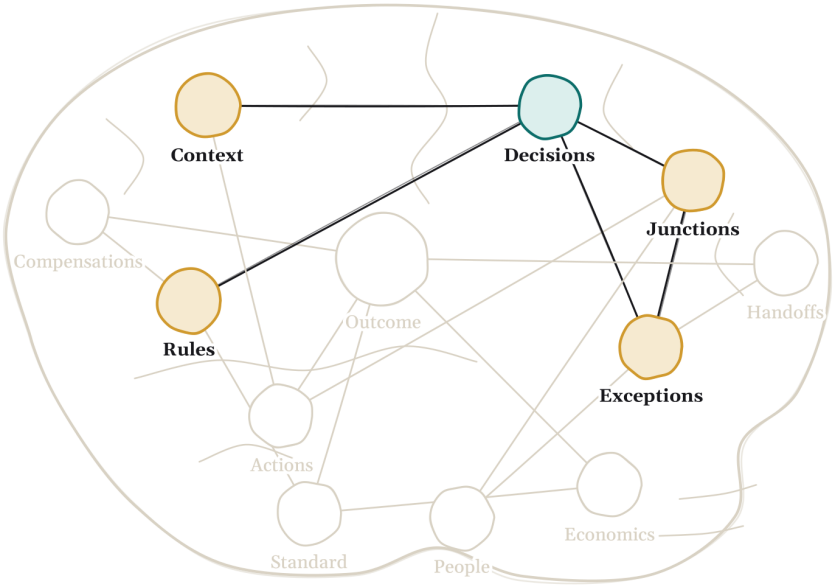
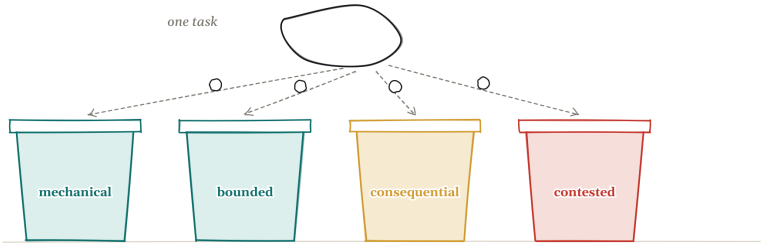


PLATE 29 Decisions

Sort every decision: mechanical, bounded, consequential or contested.

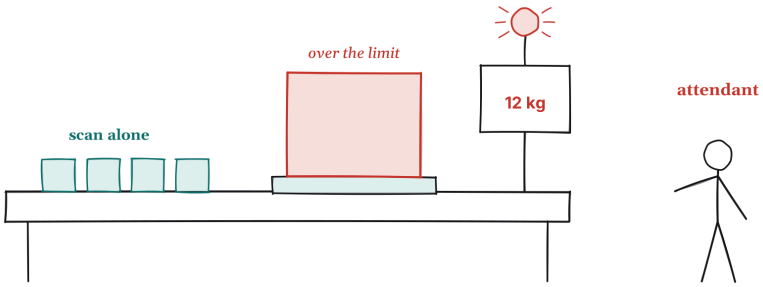
This is the node that most distinguishes the canvas from a process map. A process map shows tasks. This node lists the decisions inside them and sorts each one into four classes.



Like a recycling station with four bins: *each task is emptied out and every decision inside it goes into one of four classes.*

Mechanical decisions follow a rule with no judgment: is the policy in force, is this a duplicate, are the documents complete. They can run alone. **Bounded** decisions involve judgment within a written threshold: repair or replace below a set amount, a reserve within a range. They can run alone against the threshold, with cases outside it sent to a person. **Consequential** decisions carry enough weight that a person always makes them, whatever the evidence: declining a claim, referring someone for fraud. **Contested** decisions are the ones the organization never actually agreed on.

The classes tend to follow the distribution of cases. Mechanical and bounded decisions live in the body, where cases are many and similar. Consequential decisions cluster in the tail, where cases are rare, large and different from each other. A decision that looks bounded on the average case can turn consequential on a large one, so set thresholds by value as well as by type.

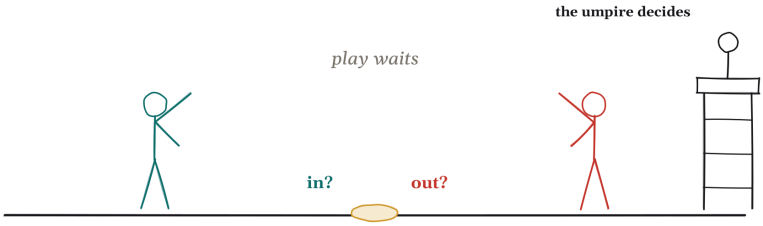


Like a self-checkout: *small items scan through on their own, but anything over the weight limit calls the attendant, whatever kind of item it is.*

Sort the decisions in the room, with the process owner, an experienced operator and the system owner present. Disagreement about which class a decision belongs to is useful information. Usually it means the decision is less settled than everyone assumed.

The node is signed when every decision is sorted and the contested ones are at zero. That last condition is strict on purpose. A contested decision cannot be engineered away. It has to be settled by someone with the authority to settle it, before the loop acts alone. Part Four gives it a chapter of its own.

In the claims loop, the sort found four mechanical decisions, four bounded, two consequential and one contested. The contested one was whether goodwill payments are allowed on borderline cover. It went to the claims committee. Until the committee decides, the node stays at Draft.



Like a disputed line call in tennis: *the players can argue, but only the umpire can settle it, and play waits until the call is made.*

XXV

Junctions

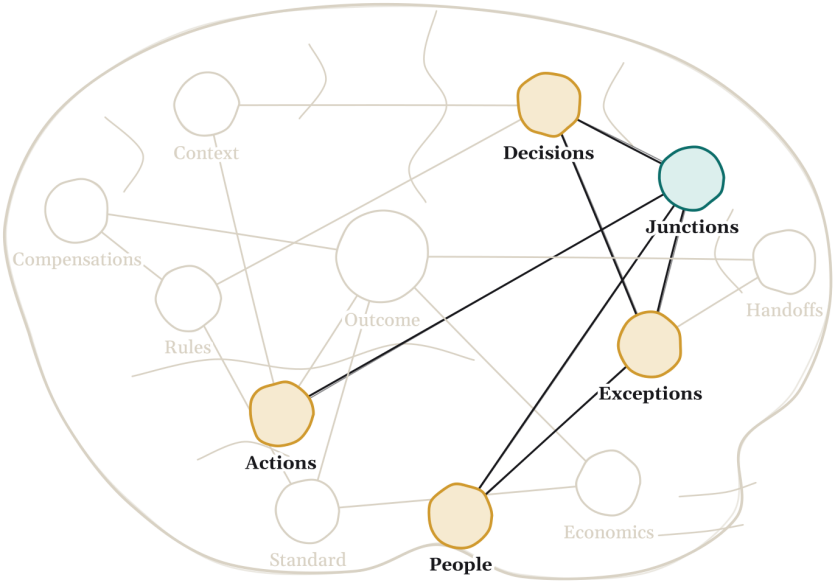
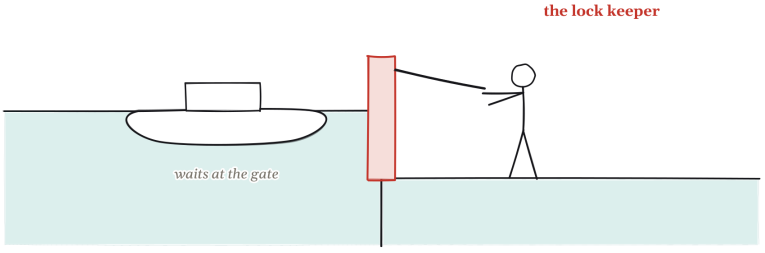


PLATE 30 Junctions

Where does a person decide, what do they see, and how is an action reversed?

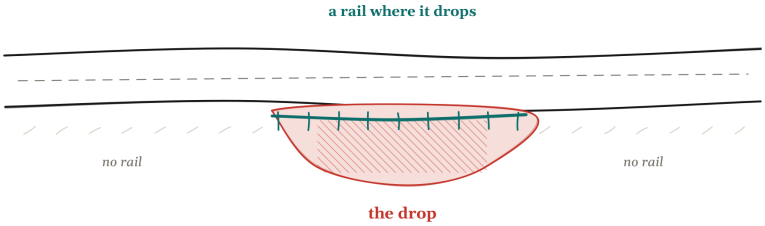
A junction is Toyota's stop cord rebuilt for an executor that decides. It is a defined point where the loop pauses and a named person decides. Every consequential decision gets one. So does every action that cannot be undone or costs a lot when wrong.



Like a canal lock: *the boat stops at a set point, the lock keeper looks and decides, and only then does the gate open.*

Place junctions by reversibility and the cost of an error, not by comfort. In a power-law domain, that usually means placing them in the tail, where a few cases carry most of the risk. Teams new to agents tend to put a person on every step, which recreates the second review the compensations node just removed. Teams that are overconfident put a person on none. The canvas asks for a reason for each junction, and the reason has to be about consequences.

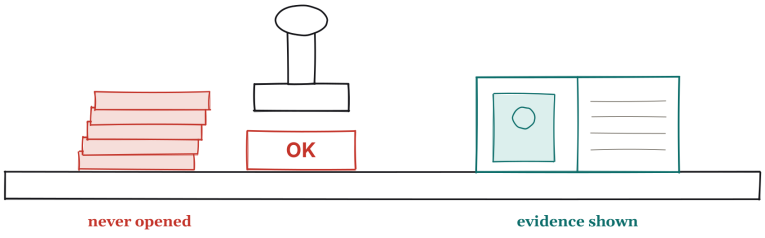
For each junction, write down three things: who decides, what evidence they see and how the action is reversed if it was wrong. Then name who can pause the whole loop. That is the andon cord at the level of the loop, and someone must hold it.



Like guard rails on a mountain road: *they go where the drop is steep, not along every straight and not nowhere, and each one has a reason.*

The node is signed when every irreversible or high-cost action passes a named junction. The usual failure is a junction with a name but no evidence. A person asked to approve an agent's decision without seeing why it was made becomes a rubber stamp within a week.

In the claims loop, three junctions were placed. Declining a claim goes to a senior adjuster, who sees the rule trace and evidence and can reopen it within thirty days. Any amount over fifteen thousand dollars goes to a team lead. A fraud referral goes to an investigations analyst. The claims operations lead can pause the loop.



Like a border officer stamping passports without opening them: *a junction that shows no evidence turns the named person into a rubber stamp.*

XXVI

Exceptions

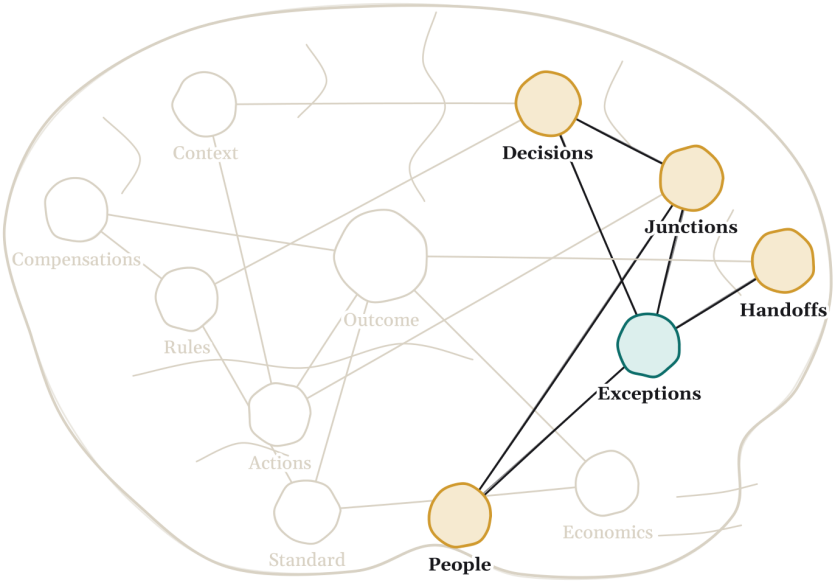
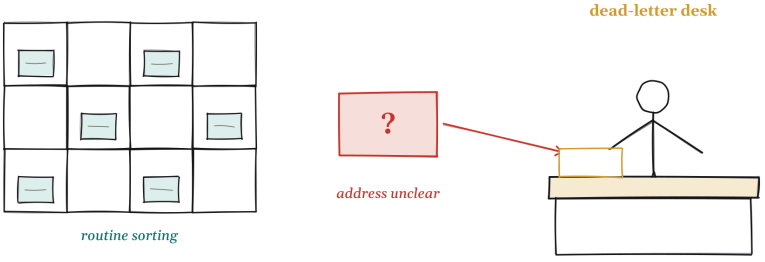


PLATE 31 Exceptions

Where do the cases the loop cannot handle go, and who works that queue?

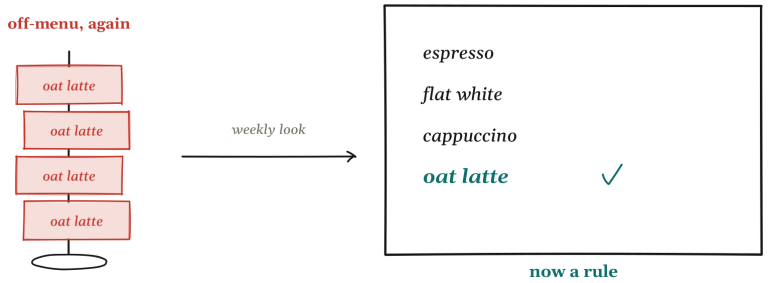
Every loop meets cases its rules do not cover: missing evidence, unusual circumstances, a policy written before a rule existed. The exceptions node gives those cases a defined place. Without one, they pile up in inboxes, or worse, the executor makes something up.



Like a post office's dead-letter desk: *a letter with an unreadable address goes to one named counter, instead of piling up or being dropped at a guessed house.*

Name the queue, its owner and a time limit. Decide what the customer sees while their case waits, because a long silence is a service failure even if the eventual decision is right. Then set a review rhythm. Once a week, someone looks at the exceptions and asks which ones repeat. A repeated exception is usually a missing rule. Writing that rule moves the case back into the routine.

The exception queue is also where shifts in the distribution show up first. New kinds of cases land here before anyone has a name for them. Watch the queue's mix as well as its size.

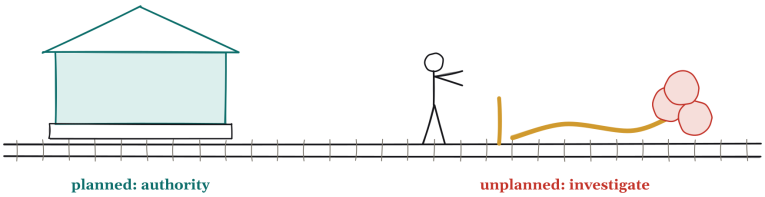


Like a café where the same off-menu order keeps coming in: *once someone notices it repeating, it gets written onto the menu and becomes routine.*

This node and the junctions node are related but different. A junction is a planned stop for a known kind of decision. An exception is an unplanned stop for a case the design did not foresee. Both need people, but the skills differ. Junctions need authority. Exceptions need investigation.

The node is signed when the queue has an owner, a time limit and a weekly review. The usual failure is an exception queue that becomes the old process under another name, with most cases ending up there. If more than a small share of cases are exceptions, the rules node is incomplete.

In the claims loop, cases outside the rules or missing evidence go to the adjuster queue. The assessment team lead owns it, with a limit of forty-eight hours. A weekly review turns repeated exceptions into rules.



Like a train's scheduled station stop beside an unscheduled stop for a fallen tree: *one needs someone with authority, the other someone who can investigate.*

XXVII

Handoffs

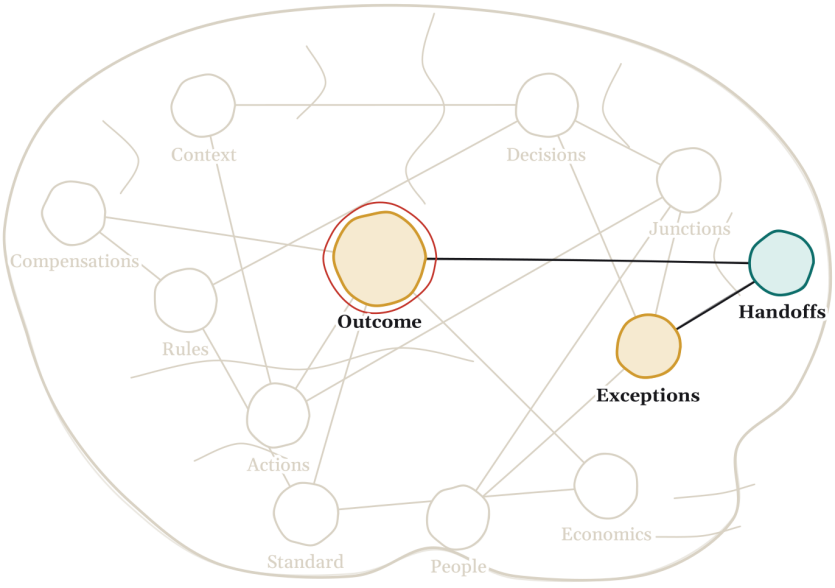
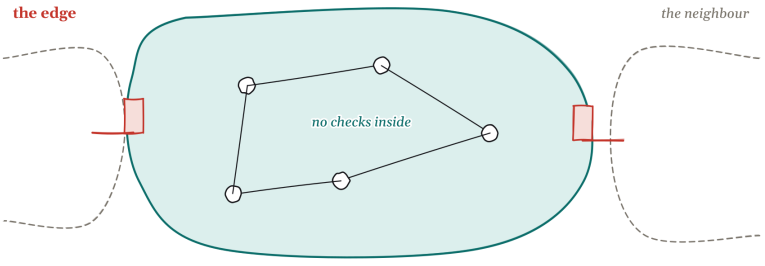


PLATE 32 Handoffs

What does this loop receive, and what does it pass on?

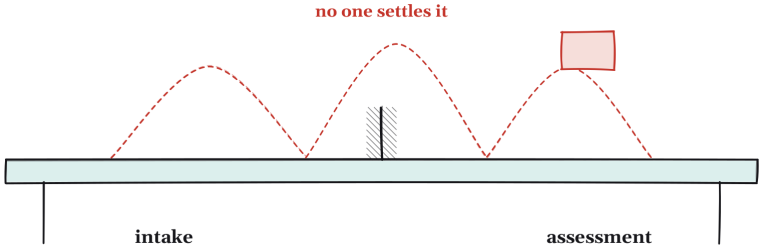
Ford made the handoff the unit of industrial work. In the work loop it moves to the edge. Inside the loop, a single owner and a single design remove most internal handoffs. What remains are the boundaries between this loop and its neighbours.



Like a free-travel zone: *inside it you drive from town to town with no checks, and the border posts sit only where it meets its neighbours.*

For each boundary, write down what the loop receives, from which loop, and in what state. Then write down what it passes on, to whom, and in what state. Finally, name who resolves a mismatch. Mismatches are where loops fail quietly. Intake sends a claim it believes is complete. Assessment finds a document missing. Without a named owner for the boundary, the claim bounces between teams.

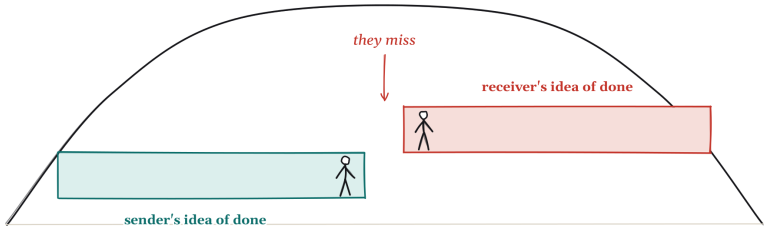
Handoffs are also where the Distribution model meets the loop. A handoff to a customer, a partner or a regulator carries more risk than a handoff to another internal loop, and deserves a stricter state definition.



Like a ball in table tennis: *with nobody named to settle a mismatch at the boundary, a claim just bounces back and forth between intake and assessment.*

The node is signed when every handoff names the state passed and who resolves a mismatch. The usual failure is defining handoffs from one side only. The sending loop's idea of complete is rarely the receiving loop's idea of complete. Both owners have to agree.

In the claims loop, assessment receives a validated claim file from intake. It passes an assessed claim with an amount and rationale to settlement, and suspected fraud to investigations. At the time of the worked run, nobody had been named to resolve mismatches with intake, so the node stayed at Draft.



Like a tunnel dug from both ends: *unless both crews agree where they will meet, the two halves miss each other in the middle.*

XXVIII

Standard

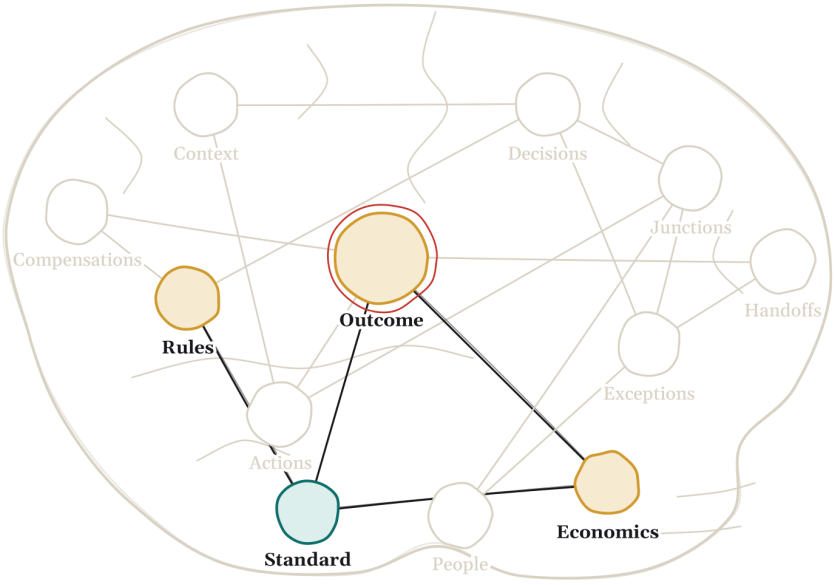
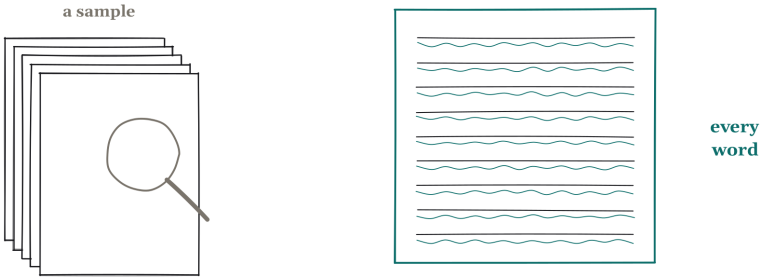


PLATE 33 Standard

How is every outcome graded?

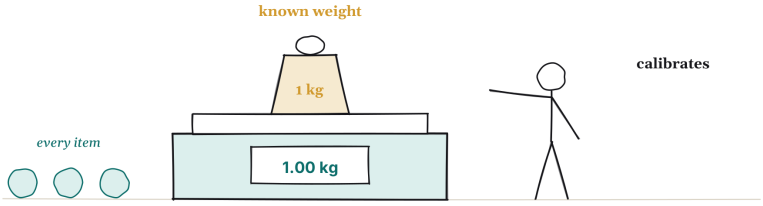
Taylor wrote the standard so people could follow it. The canvas writes it so every result can be checked. The difference is coverage. Traditional quality control checks a sample, because checking everything costs too much when people do the checking. With agents in the loop, every outcome can be graded, and it should be.



Like a spell checker beside a proofreader: *the proofreader samples a few pages, while the checker reads every word of every page.*

Write the acceptance criteria in plain language first. Then decide who or what grades each criterion: a rule, a model or a person. Rules suit criteria that can be checked mechanically, such as whether an amount is within policy limits. Models suit criteria that need reading, such as whether a rationale actually traces to the rules. People calibrate the graders, by auditing a small share of graded outcomes and correcting the graders when they drift.

Grade the tail more closely than the body. The audit share that calibrates the grader can be small for routine cases and much larger for the few cases that carry most of the value.

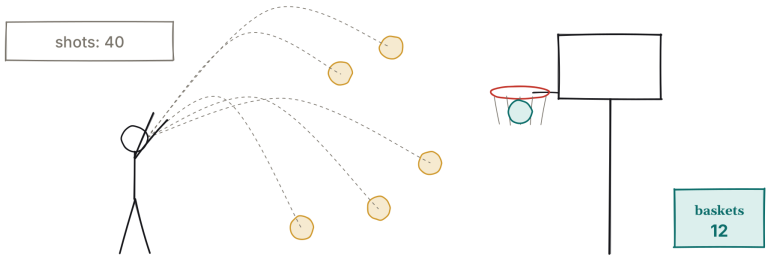


Like shop scales tested with a known weight: *the scales weigh every item, and a person checks them now and then so their readings stay true.*

The standard is also the loop′s early-warning system. A falling acceptance rate shows a problem before customers complain. It is the measure the economics node depends on, because cost per accepted outcome needs a definition of accepted.

The node is signed when every outcome is graded against a written standard. The usual failure is measuring activity instead of acceptance: claims processed per hour, tasks automated, agents deployed. None of those says whether the results were right.

In the claims loop, an assessment is accepted when the amount is within policy terms, the rationale traces to the rules and the customer letter is correct. A rules engine checks the terms. A model checks the rationale. A human audit of two percent of claims calibrates the model.



Like a basketball player counting shots taken: *the activity number climbs, but only baskets made, the accepted outcomes, show whether the work was right.*

XXIX

People

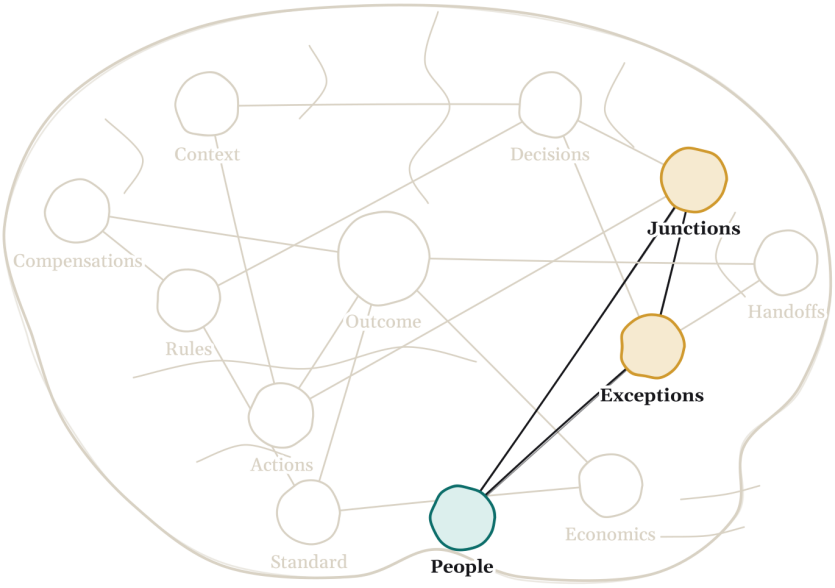
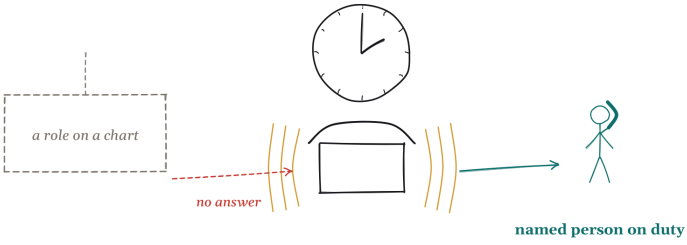


PLATE 34 People

Who answers for the loop, and whose work changes?

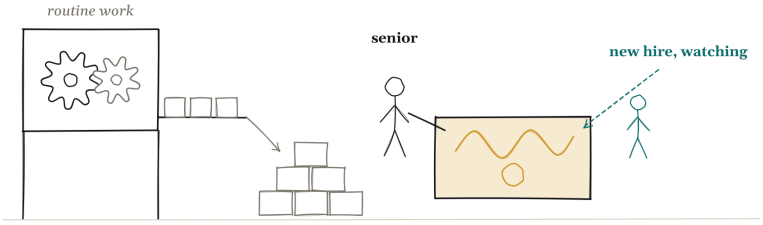
Sloan's lesson was that an outcome needs an owner. This node names one for the loop as a whole, and names the people who hold each junction. Names, not roles, because a role on a chart does not answer the phone at two in the morning.



Like a phone ringing at two in the morning: *a named person on duty picks it up, while a box on an org chart never answers.*

The node also asks a harder question: whose work changes, and how? In the claims loop, adjusters move from assessing every file to working junctions and exceptions. That is a real change in what their day looks like, and it should be said before launch, not discovered after it.

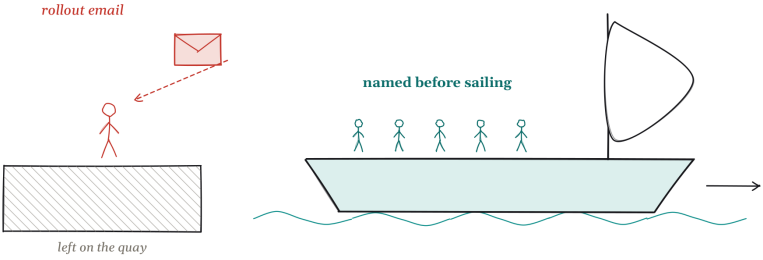
The last question is about learning. Experienced adjusters learned their judgment by assessing thousands of routine files. If agents now handle the routine files, new adjusters lose the work that used to train them. The loop needs a deliberate path for that, such as shadowing junction decisions or reviewing a sample of agent assessments with a senior colleague.



Like a stonemason's yard where a machine now cuts the plain blocks: *the apprentice must be set beside the master on the carved stones, or never learns.*

The node is signed when named people are in place before production. The usual failure is treating this node as a communications task to handle after the design is done. People who learn about a changed role from a rollout email rarely help make the loop work.

In the claims loop, the Head of Claims Operations owns the loop. Junction holders are named for all three junctions. How junior adjusters will learn was still open at the time of the worked run, so the node stayed at Draft.



Like a crew mustered on deck before the ship sails: *whoever is told by letter after departure is left on the quay and helps nobody sail.*

XXX

Economics

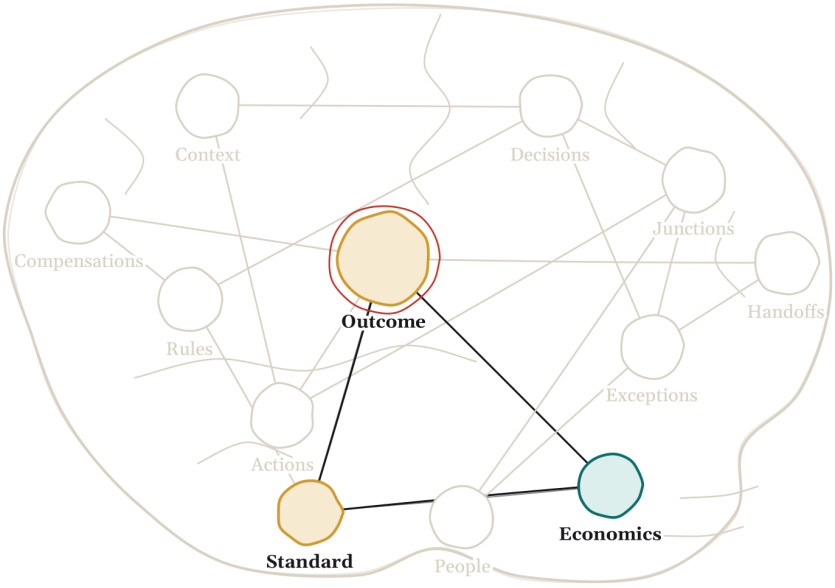
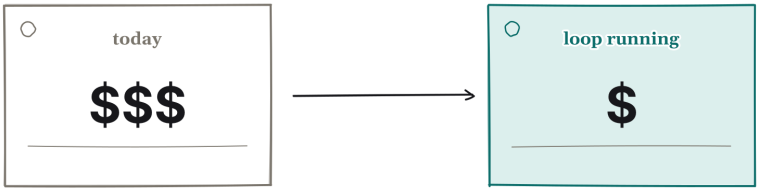


PLATE 35 Economics

What does one accepted outcome cost, compared with the measured baseline?

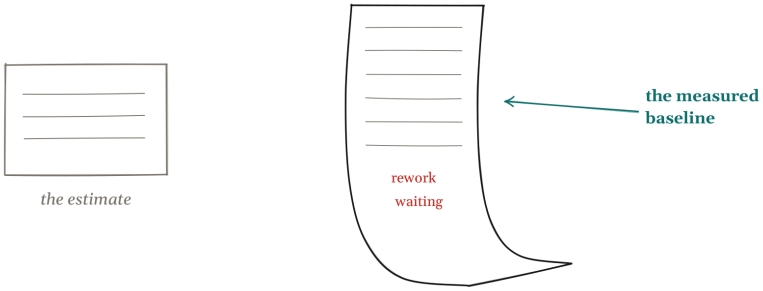
The last node turns the loop into numbers the Financial model can use. It compares the cost of one accepted outcome today with the cost once the loop runs, along with cycle time and the share of outcomes right the first time.



Like a unit-price label on a shelf: *compare the cost of one accepted outcome today with its cost once the loop runs, not the size of the bill.*

Measure the baseline before changing anything. Estimates do not count, because every estimate of today's cost is too low. It leaves out the rework, the waiting and the time experienced people spend answering questions. Then set a target for each measure.

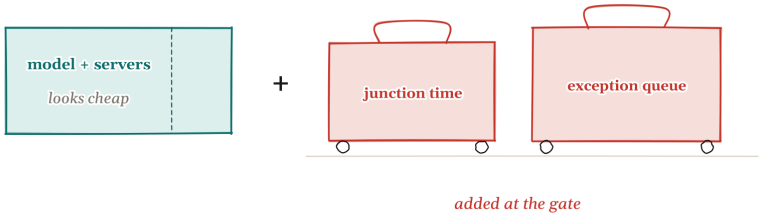
Measure by segment of the distribution, not only on average. A loop can lower the average cost per claim while raising losses on the large claims, and an average will hide that. Report the body and the tail separately.



Like a till receipt that runs far longer than the shopping list: *measure today's real cost, rework and waiting included, before trusting any estimate.*

When calculating the cost of the new loop, include everything. That means model and infrastructure costs, plus the time people spend at junctions and in the exception queue. A loop that looks cheap because its people costs were left out will look expensive the first time someone adds them back.

The node is signed when cost per accepted outcome is below the baseline, measured rather than estimated. Until the loop has run in shadow, this node can only be a draft. That is expected.



Like a budget airline fare: *the ticket looks cheap until the bags are added at the gate, so count the people time at junctions from the start.*

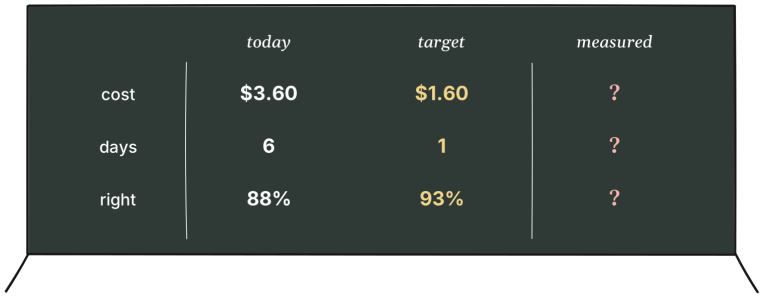
In the claims loop, the measured baseline was $3.60 handling cost per claim, six days in assessment and 88 percent assessed right first time. The targets were under $1.60, one day and 93 percent. The run cost was set to include

junction time and the exception queue, and had not yet been measured in shadow.

PART FIVE

From One Loop to the Firm

A single canvas describes one loop. A firm runs dozens. This part covers the decision that blocks most loops, the map that places all of them, the gates that pace their growth and the session that fills a canvas in an afternoon.



Like a training-ground chalkboard: *today's times are written up, the targets chalked beside them, and the trial column stays empty until the shadow run.*

XXXI

The Contested Decision

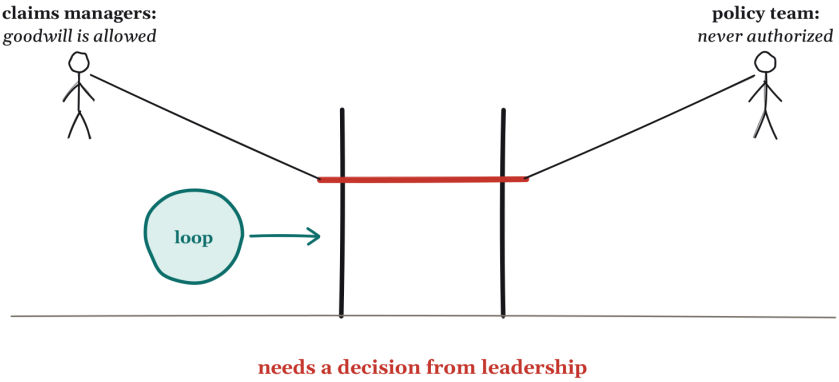
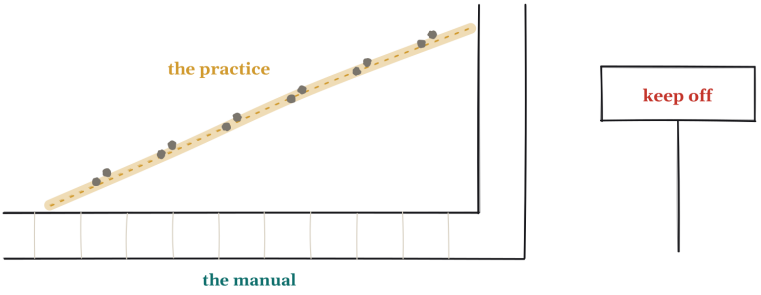


PLATE 36 The Contested Decision

Sorting decisions in a room almost always turns up at least one that nobody can classify. The manual says one thing, the system does another, and two senior people give different answers. That is a contested decision.

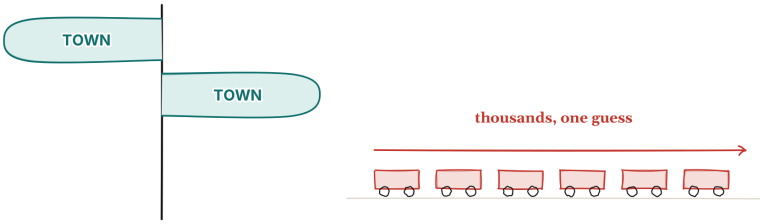
Contested decisions are not rare, and they are not signs of a badly run firm. They are what happens when a business grows faster than its rules. Two teams solve the same problem in different ways. A policy written for one market is applied in another. A manager makes a sensible exception, then another, until the exception is the practice. Experienced staff absorb the difference case by case, and nobody needs to settle it.



Like a footpath worn across a lawn beside the paved path: *a sensible shortcut, taken again and again, becomes the real route though no plan shows it.*

An agent cannot absorb it. Given a contested decision, it will either act on a guess or stop. Both look like a technology failure, though the cause is organizational. **No amount of engineering can resolve a contested decision.** It needs a choice from someone with the authority to make it, written down and owned.

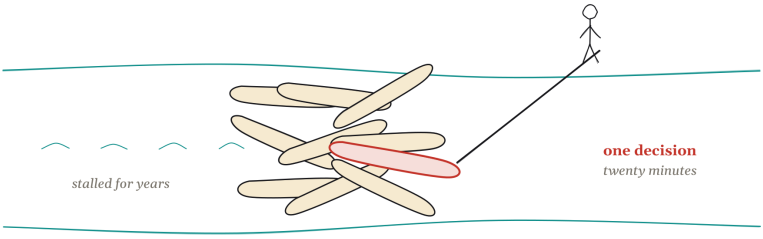
The canvas treats contested decisions as the first thing to resolve, not the last. They hold the decisions node at Draft, and that holds the loop below the Bounded gate. This is deliberate. A loop that acts alone while a contested decision is open will make that decision thousands of times, in whichever direction its guess falls.



Like a signpost pointing both ways to the same town: *a driver can ask, but a machine either stops or sends every car down whichever arm it guessed.*

Resolving one is usually quick once it is visible. The goodwill question in the claims loop took one committee meeting. The committee decided that goodwill payments are allowed up to a set amount, on claims within thirty days of the cover lapsing, with a team lead's approval. That decision created a new bounded rule and a new junction. It took twenty minutes. It had been open for years.

Many stalled AI programs have a few of these at their centre. The canvas makes them visible on the first afternoon, before anyone writes a line of code. For many firms that alone justifies the exercise.



Like a log jam on a river: *years of stalled flow, freed in twenty minutes once someone with a pole pulls out the one key log.*

XXXII

The Work Loop Map

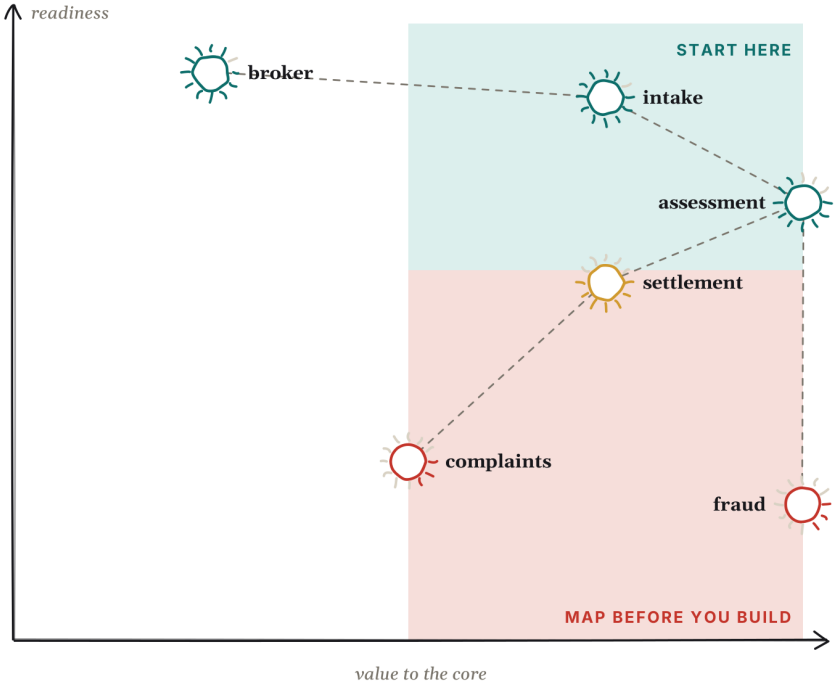
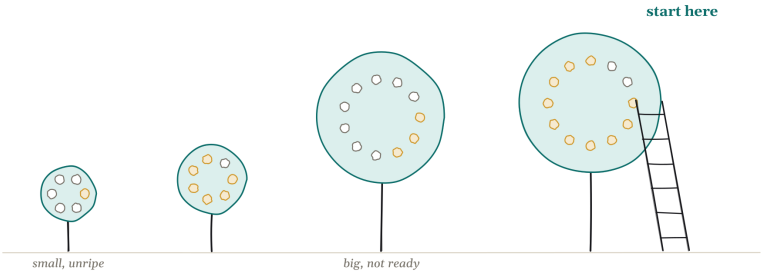


PLATE 37 The Work Loop Map

A firm runs many loops, and they do not all deserve attention at once. The Work Loop Map places each loop on two axes.

The horizontal axis is value to the core of the business, scored from one to five. A loop scores high when it touches customers directly, carries significant cost or risk, or produces the firm's main product. It scores low when it supports other internal work. The vertical axis is readiness, read directly from how many of the loop's nodes are signed.

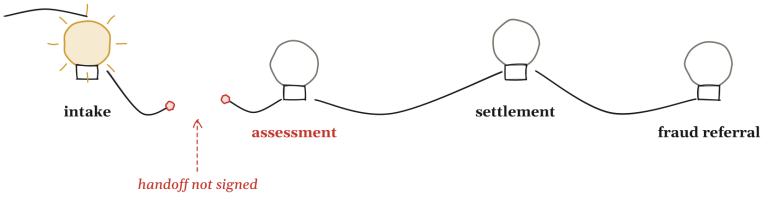


Like a fruit picker sizing up an orchard: *each tree scored by its crop and its ripeness, with the ladder going up the big ripe tree first.*

The map divides into four regions. Loops in the top right matter and are ready, so that is where to start. Loops in the bottom right matter but are not ready, so the work there is mapping, not building. Loops in the top left are ready but less important. They make good practice for a team learning the method. Loops in the bottom left can wait.

The map also draws the handoffs between loops. That turns a list of projects into a picture of the organization's wiring. It shows which loops depend on each other, and therefore which ones cannot reach the core alone. In the insurer's map, assessment depends on intake and feeds settlement and fraud referral. Assessment cannot reach the Core gate until its handoffs with intake are signed, however good the loop is on its own.

Seen this way, the map is a picture of the firm's connectome: its loops and the connections between them. It is also the most useful page to show a board. It answers three questions that most AI programs cannot: where the firm is starting, why there, and what has to be true before the next loop moves.



Like a string of lights wired one to the next: *the assessment bulb cannot glow, however good, until its wire from intake is connected.*

XXXIII

The Scale Gates

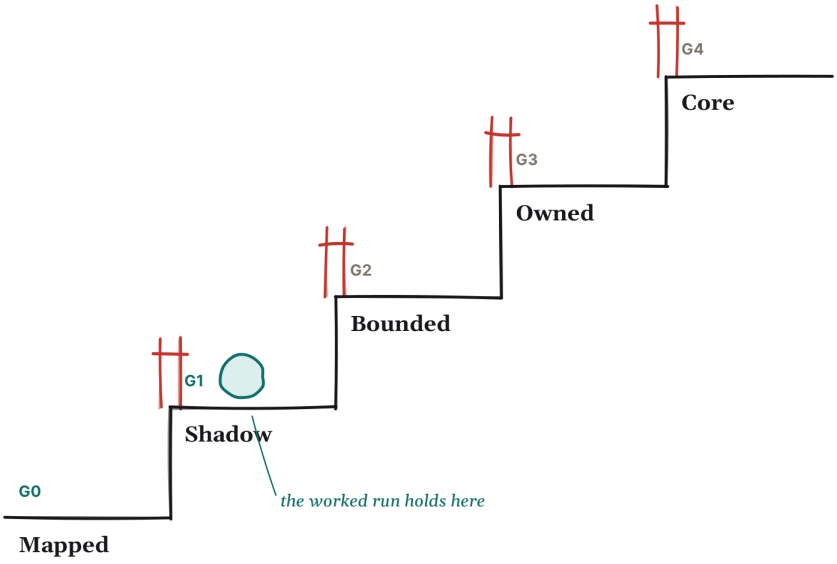
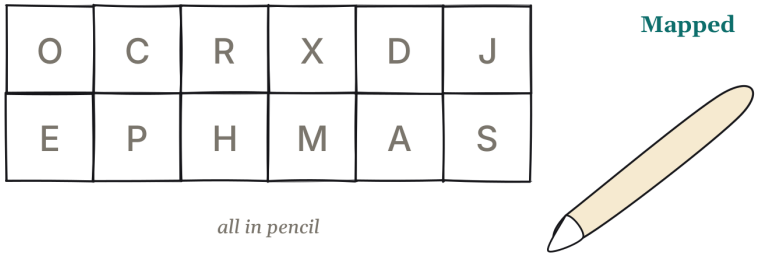


PLATE 38 The Scale Gates

Readiness should not be a feeling. The Work Loop Canvas ties it to five gates, and each gate opens only when specific nodes are signed.

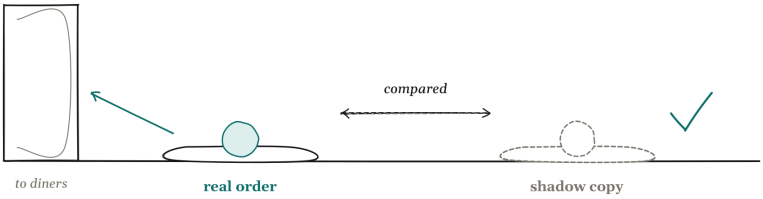
The first gate is **Mapped**. All twelve nodes have at least a draft. The loop is described but not yet agreed.



Like a crossword filled in pencil: *every one of the twelve squares has a guess, so the loop is Mapped, but none is inked and agreed yet.*

The second is **Shadow**. The loop runs beside today's process without writing to any system. Every outcome is graded against the standard, and the results are compared with what people decided. Shadow requires the outcome, compensations, rules, context and standard to be signed. It is the cheapest place to learn, because nothing the loop does reaches a customer.

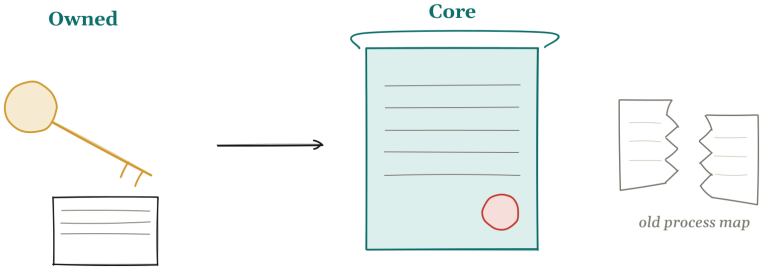
The third is **Bounded**. The loop acts alone on mechanical and bounded decisions, with junctions and the exception queue live. It adds signed actions, decisions, junctions and exceptions. This is the gate a contested decision blocks.



Like a trainee cook plating a copy of every order that never leaves the kitchen: *compared with the chef's plate, it costs nothing if it goes wrong.*

The fourth is **Owned**. The loop runs in production with a named owner, at a cost per accepted outcome below the measured baseline. It adds signed people and economics.

The fifth is **Core**. The loop writes to systems of record and hands off to neighbouring loops. The old process map is retired, and the canvas becomes the description of how the work is done. Core requires all twelve nodes, including handoffs.

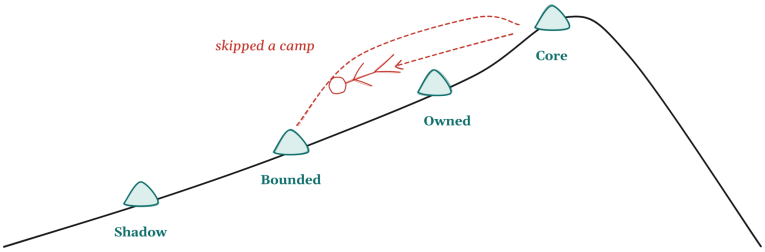


Like a tenant who becomes the owner: *first the keys and the bills in their own name, then the deeds rewritten and the old lease torn up.*

A loop moves up a gate only when the nodes that gate depends on are signed. Scale follows signatures, not confidence. A pilot that jumps a gate

has borrowed against the core. It usually repays the loan in production, as an incident that sets the whole program back.

The gates also give a program a shared language for progress. "The assessment loop is at Shadow, held by one contested decision" is a status report a board member can understand, and it says exactly what would move it.



Like a climber who moves to the next camp only after the last one:
skipping a camp borrows against the summit and is paid back as a fall.

XXXIV

Running the Session

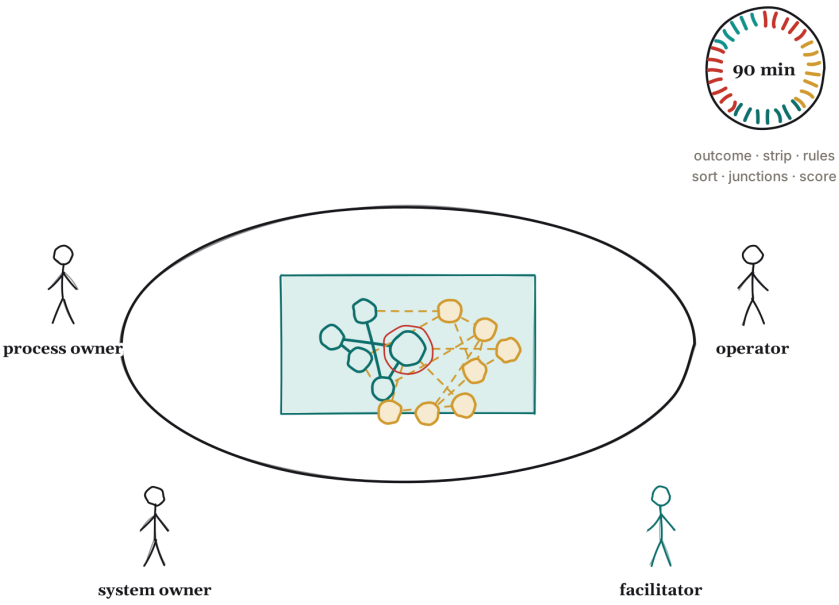
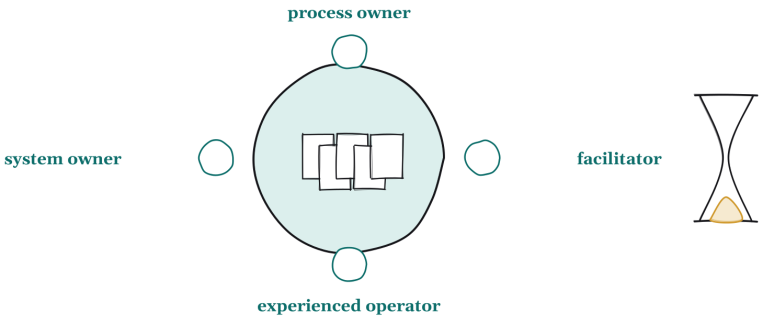


PLATE 39 Running the Session

The Business Model Canvas spread because it came with a ritual. People knew how to run a session with it. The Work Loop Canvas needs the same.

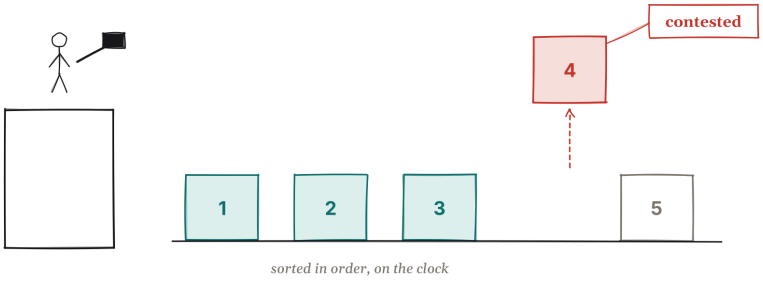
A first session takes about ninety minutes and needs four people. The process owner answers for the outcome and makes calls. An experienced operator, someone who does the work every day, knows the unwritten rules and the real exceptions. The system owner knows what the systems actually do and what the configuration says. A facilitator keeps time and holds the order of the nodes.



Like a game of bridge: *always four at the table, each seat with its own job, and a ritual everyone knows for playing the hand.*

Fill the nodes in number order. Spend the first fifteen minutes on the outcome and do not move on until it is agreed. Spend the next twenty on compensations, walking the current process step by step. Rules and context take another twenty, and the list of rules will not be finished in the room. Capture what people know and assign owners to complete it.

The decision sort is the heart of the session. Give it twenty minutes and expect disagreement. When people disagree about a decision′s class, write it down as contested and move on. Do not try to resolve it in the room unless the person with authority is present.

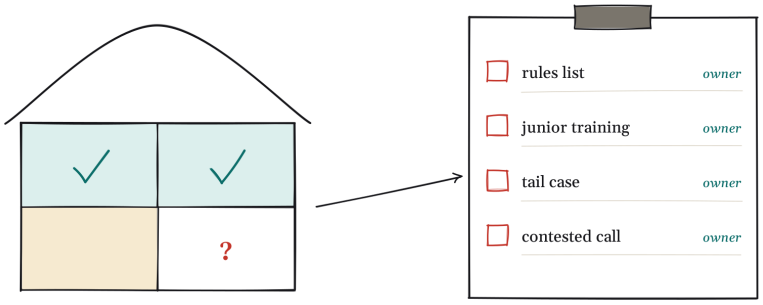


Like an auction run lot by lot to the clock: *a lot nobody agrees on is tagged and set aside for someone with authority, and the sale moves on.*

Make sure one of the cases on the table comes from the tail of the distribution: a large, unusual or expensive case. Routine cases show the steps. The tail case shows where judgment lives.

Use the last fifteen minutes for junctions, people and economics, at draft level. Close by scoring every node as Blank, Draft or Signed, and reading out the gate the loop has reached and what holds it there.

A first session rarely signs more than half the nodes. That is fine. Its real output is the list of what is missing, with an owner beside each item. Most teams reach Shadow within two or three weeks of the first session, once the rules list is complete and the contested decisions have gone to someone who can settle them.



Like a snag list after walking a new house: *half the rooms pass, and the real output is the list of what is missing, each with an owner.*

XXXV

What the Canvas Leaves Out

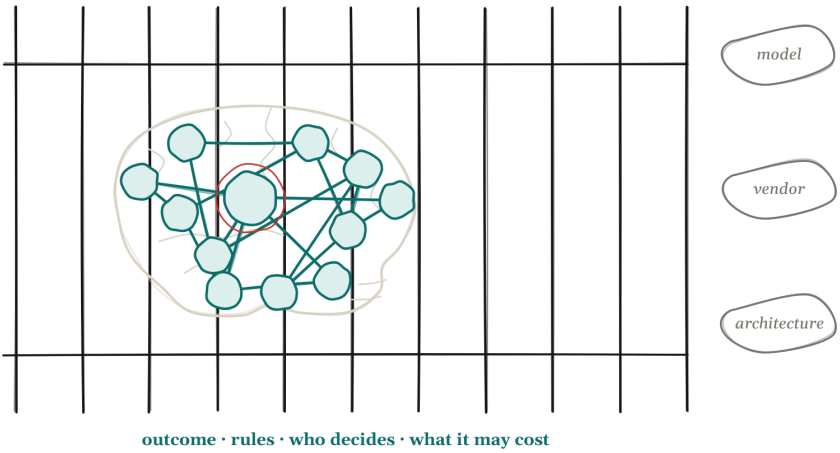
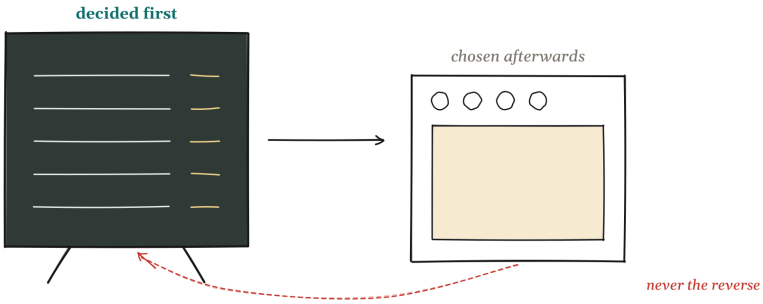


PLATE 40 What the Canvas Leaves Out

The Work Loop Canvas deliberately says nothing about which model to use, which vendor to buy from or how to architect the system. Those choices matter, and they come afterwards.

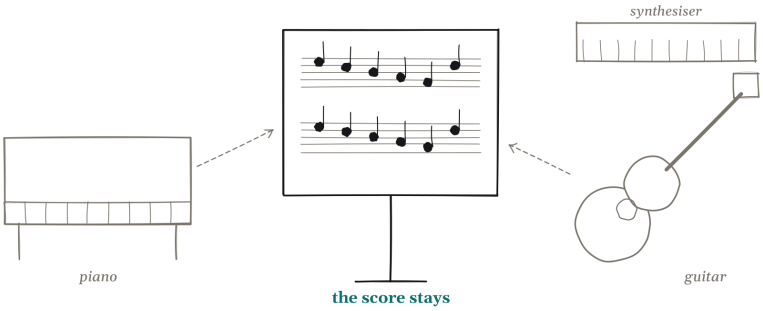
The canvas records the decisions only the business can make: what counts as a good result, which rules apply, who decides what and what it may cost. Implementation follows from those answers. A team that starts from the technology tends to make the business decisions implicitly, inside configuration files, where nobody signs them. That is the same mistake enterprise software made in the 1990s, repeated with a more capable executor.



Like writing the menu before buying the oven: *the business picks the dishes and prices first, or the oven ends up deciding the menu.*

Leaving the technology out also keeps the canvas durable. Models, vendors and architectures will change many times over the life of a loop. The outcome, the rules, the decisions and the people change much more slowly. A canvas written in business terms survives a change of model. A design written in technical terms has to be redone.

The test for a finished canvas is simple. **Someone who cannot name a single AI model should be able to read it and sign it.**



Like a page of sheet music: *the piano, the guitar and the synthesiser come and go, while the score, written in plain notes, stays the same.*

The Six Moves

The method compresses into six moves, in order.

Know the distribution. Before anything else, draw the spread of the loop's cases and find where the volume sits and where the money sits.

Strip the workarounds. Before writing a rule or placing an agent, strike every step that exists only to work around a constraint that no longer applies.

Write down the rules. Bring every rule into one list, from documents, configuration and people's heads, with an owner and a date. List the contradictions as findings.

Sort the decisions. Put every decision in one of four classes. Send the contested ones to someone who can settle them before the loop acts alone.

Place the junctions. Put a named person wherever an action cannot be undone or an error costs a lot, and say what they see and how they reverse it.

Sign the loop. Name an owner, grade every outcome, measure the baseline and move the loop up one gate at a time.

The Meters

A program built on work loops should report a small set of measures, and avoid a larger set of tempting ones.

Report these for each loop. Start with the share of steps struck as compensations, the number of rules with an owner and the number of open contradictions. Add the count of decisions in each class, with contested decisions heading to zero. Track the number of junctions, the share of cases that reach one, and the size and age of the exception queue. Finish with the acceptance rate against the standard and cost per accepted outcome against the measured baseline, each reported separately for the body and the tail of the distribution. At the level of the firm, report how many loops sit at each gate.

Avoid these: tasks automated, agents deployed and percentage of steps covered by AI. Each of them can rise while the loop gets worse. They measure activity, and the canvas measures acceptance.

Epilogue: The Executor Changes

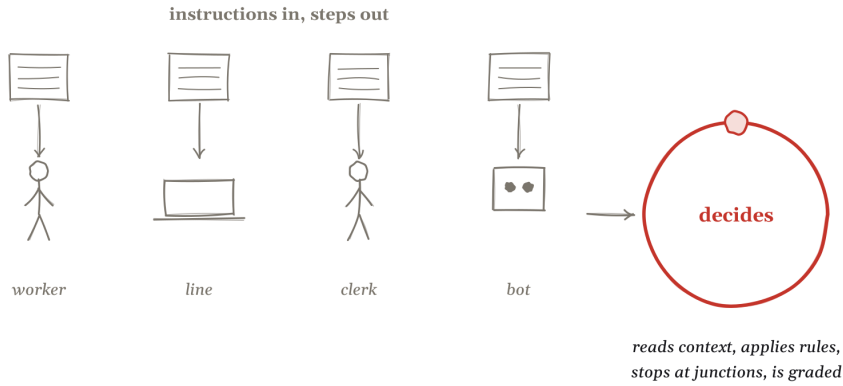


PLATE 41 The Executor Changes

Every generation of managers designed work around the executor it had. Taylor's worker, Ford's line, Sloan's divisions, Toyota's team, the ERP clerk and the software robot all followed instructions. The intelligence lived in the design, and the executor carried it out.

The executor has changed. It reads context, interprets cases and makes decisions within bounds. That does not make the old lessons obsolete. It makes them all necessary at once. The standard, the handoff, the owner, the stop cord, the stripped workaround and the written rule were each invented to solve one problem. An executor that decides needs every one of them, in one design, around one outcome.

That design is a work loop, and a firm is a set of them wired together. The business model told a firm what it sells. The work loops tell it how its work gets done. In the 2000s, the firms that drew their business model clearly moved first. In this era, the firms that draw their loops clearly will.

Strip the workarounds. Sort the decisions. Sign the loop.

Appendix A: The Blank Canvas

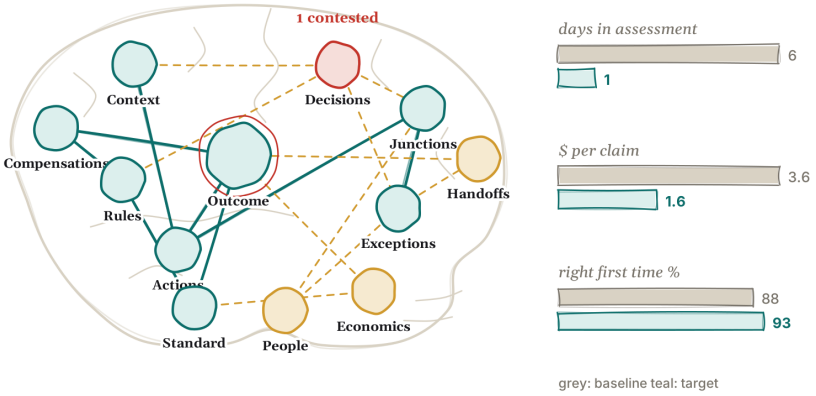
The twelve nodes, in fill order, with the question each one asks and the test for when it is signed. A node is Blank until something is written, Draft once it is written, and Signed once it is agreed and owned by a named person.

#	Node	The question	Signed when
1	Outcome	What does one accepted result look like, and who receives it?	Someone outside the team can tell an accepted result from a rejected one.
2	Compensations	Which steps exist only to work around a constraint that no longer applies?	Every remaining step is required by the outcome itself.
3	Rules	Where does each rule live today, and who owns it?	Every rule has an owner and a date, and contradictions are listed.
4	Context	What must the executor read to act well?	Each source has a keeper, and the executor can reach it with the right permissions.
5	Actions	What can the executor change, and in which systems?	Every write is listed with its system, its permission and whether it can be undone.
6	Decisions	Sort every decision: mechanical, bounded, consequential or contested.	Contested decisions are at zero.
7	Junctions	Where does a person decide, what do they see, and how is an action reversed?	Every irreversible or high-cost action passes a named junction.
8	Exceptions	Where do cases the loop cannot handle go, and who works that queue?	The queue has an owner, a time limit and a weekly review.
9	Handoffs	What does this loop receive, and what does it pass on?	Every handoff names the state passed and who resolves a mismatch.
10	Standard	How is every outcome graded?	Every outcome is graded against a written standard.
11	People	Who answers for the loop, and whose work changes?	Named people are in place before production.
12	Economics	What does one accepted outcome cost, compared with the measured baseline?	Cost per accepted outcome is below baseline, measured rather than estimated.

The Scale Gates and the nodes each one requires:

Gate	What the loop may do	Nodes that must be signed
G0 Mapped	Nothing yet; described, not agreed	All twelve at Draft or better
G1 Shadow	Run beside today's process with no writes; grade every outcome	Outcome, Compensations, Rules, Context, Standard
G2 Bounded	Act alone on mechanical and bounded decisions, junctions live	Adds Actions, Decisions, Junctions, Exceptions
G3 Owned	Run in production below the measured baseline	Adds People, Economics
G4 Core	Write to systems of record and hand off to other loops	All twelve, including Handoffs

Appendix B: The Worked Run, Filled



38 rules · 7 contradictions · 3 junctions · gate G1 Shadow

PLATE 42 A Worked Run: Claims Assessment

The claims assessment loop of the composite general insurer, as filled at the end of its first two sessions. The firm and its numbers are illustrative. The status of each node is shown in brackets.

1. Outcome [Signed]. A claim assessed, with liability and amount decided and a rationale traceable to the rules, ready for settlement. About 4,000 claims a month. Distribution: about 80 percent of claims are under $5,000 and carry about a quarter of the money paid; the largest 3 percent carry about two-fifths. Assessment takes 6 of the claim′s 11 days today. Received by the settlement team.

2. Compensations [Signed]. Struck: the overnight re-score, kept because the rating system once ran only at night; a second review of every claim under $2,000, kept to catch re-keying errors that no longer happen; the paper file transfer from intake. Kept: site inspection for structural damage.

3. Rules [Signed]. 38 rules: 21 in the claims manual, 9 in rating-engine settings, 8 only in senior adjusters′ heads. 7 contradictions, for example water

damage covered after 30 days unoccupied in the manual and 45 in the engine. Owner: Head of Claims Policy.

4. Context [Signed]. Policy and endorsements; claim notes and photos; repair estimates; prior claims on the policy; supplier price list, kept by procurement and refreshed monthly. Inspection reports arrive scanned and are converted by operations on arrival.

5. Actions [Signed]. Reads policy administration, the claims system and the document store. Writes to the claims system (reserve, assessment note) and the supplier booking system. Reserve changes can be undone. A confirmed supplier booking cannot.

6. Decisions [Draft]. Mechanical 4: cover in force, deductible, duplicate check, document completeness. Bounded 4: repair or replace under $5,000; reserve setting within range. Consequential 2: declining a claim; fraud referral. Contested 1: goodwill payments on borderline cover, sent to the claims committee.

7. Junctions [Signed]. Declining a claim: senior adjuster, sees the rule trace and evidence, can reopen within 30 days. Any amount over $15,000: team lead approves. Fraud referral: investigations analyst. The claims operations lead can pause the loop.

8. Exceptions [Signed]. Cases outside the rules or missing evidence go to the adjuster queue. Owner: assessment team lead. Limit: 48 hours. A weekly review turns repeated exceptions into rules.

9. Handoffs [Draft]. Receives a validated claim file from intake. Passes an assessed claim with amount and rationale to settlement, and suspected fraud to investigations. Mismatch owner with intake not yet agreed.

10. Standard [Signed]. Accepted when the amount is within policy terms, the rationale traces to the rules and the customer letter is correct. Graded on every claim: a rules engine checks terms, a model checks the rationale, a 2 percent human audit calibrates the model.

11. People [Draft]. Owner: Head of Claims Operations. Junction holders named for all three junctions. Adjusters move from assessing every file to working junctions and exceptions. Open: how junior adjusters learn now that routine files no longer reach them.

12. Economics [Draft]. Measured baseline: $3.60 handling cost per claim, 6 days in assessment, 88 percent right first time. Target: under $1.60, 1 day,

93 percent. Run cost includes junction time and the exception queue; not yet measured in shadow.

Gate: G1 Shadow. The loop can run beside today's process now. It cannot act alone until the goodwill decision is settled, which moves Decisions to Signed. It cannot reach Owned until the shadow run measures its economics and the learning path for junior adjusters is agreed. It cannot reach Core until the intake handoff has an owner.

After the committee settled the goodwill question, the decision became a bounded rule with a new junction: goodwill payments up to a set amount, on claims within thirty days of the cover lapsing, approved by a team lead. With Decisions signed, the loop qualified for Bounded.

Appendix C: A Facilitator′s Script

A first session, ninety minutes, four people: the process owner, an experienced operator, the system owner and a facilitator.

Before the session. Pick one loop from the top right of the Work Loop Map, or the most important loop if no map exists yet. Ask the system owner to bring the configuration for the main system the loop uses. Ask the operator to bring three recent cases: one routine, one difficult and one that went wrong.

Minutes 0 to 15: Outcome. Ask for the outcome as one sentence a customer would recognize. Reject activities. Ask who receives the result and what they do next. Record volume and today′s cycle time. Do not move on until the process owner agrees.

Minutes 15 to 35: Compensations. Walk the operator′s routine case from start to finish. At every step, ask what constraint created it and whether that constraint still binds. Strike what no longer serves the outcome and note why.

Minutes 35 to 55: Rules and context. Walk the difficult case. List every rule applied and where it lives. List every source consulted. Assign an owner to finish both lists after the session.

Minutes 55 to 75: Decisions. List every decision in the three cases. Sort each into mechanical, bounded, consequential or contested. When the room disagrees, mark it contested and record who would have the authority to settle it.

Minutes 75 to 90: Junctions, people, economics and score. Draft the junctions from the consequential decisions and irreversible actions. Name the loop′s owner. Note what the baseline measures are and who will measure them. Score every node, read out the gate and list what holds the loop there, with an owner and a date for each item.

After the session. Send the canvas to everyone present within a day. Book the second session for when the rules list is complete and the contested decisions have an answer.

Notes and Sources

Frederick Winslow Taylor, *The Principles of Scientific Management* (New York: Harper & Brothers, 1911).

The moving assembly line at Ford's Highland Park plant was introduced in 1913; chassis assembly times of roughly twelve and a half hours before and about an hour and a half after are the figures commonly reported from Ford's own records. The five-dollar day was announced in January 1914. See David A. Hounshell, *From the American System to Mass Production, 1800–1932* (Baltimore: Johns Hopkins University Press, 1984).

Alfred P. Sloan Jr., *My Years with General Motors* (New York: Doubleday, 1964). Alfred D. Chandler Jr., *Strategy and Structure: Chapters in the History of the American Industrial Enterprise* (Cambridge, MA: MIT Press, 1962).

Taiichi Ohno, *Toyota Production System: Beyond Large-Scale Production* (Portland, OR: Productivity Press, 1988; Japanese edition 1978). On the automatic loom that stopped when a thread broke and the origin of jidoka, see Toyota's own historical accounts of Sakichi Toyoda's Type G loom. James P. Womack, Daniel T. Jones and Daniel Roos, *The Machine That Changed the World* (New York: Rawson Associates, 1990).

Nassim Nicholas Taleb, *The Black Swan: The Impact of the Highly Improbable* (New York: Random House, 2007), on Mediocristan and Extremistan. Daniel Kahneman and Gary Klein, "Conditions for Intuitive Expertise: A Failure to Disagree," *American Psychologist* 64, no. 6 (2009): 515–526. Gary Klein, *Sources of Power: How People Make Decisions* (Cambridge, MA: MIT Press, 1998).

Michael Hammer, "Reengineering Work: Don't Automate, Obliterate," *Harvard Business Review*, July–August 1990. Michael Hammer and James Champy, *Reengineering the Corporation: A Manifesto for Business Revolution* (New York: HarperBusiness, 1993).

SAP R/3 was released in 1992.

Alexander Osterwalder, *The Business Model Ontology: A Proposition in a Design Science Approach*, doctoral thesis, University of Lausanne, 2004. Alexander Osterwalder and Yves Pigneur, *Business Model Generation* (Hoboken, NJ: Wiley, 2010).

Andrew Campbell, Mikel Gutierrez and Mark Lancelott, *Operating Model Canvas: Aligning Operations and Organization with Strategy* (Zaltbommel: Van Haren Publishing, 2017).

Object Management Group, *Business Process Model and Notation (BPMN) Version 2.0*, 2011; *Decision Model and Notation (DMN) Version 1.0*, 2015.

The VTDF framework (Value, Technology, Distribution, Financial), the Thinking OS and the business engineering loop are set out in full in *The Business Engineer*, the Library's Foundation Volume.

The composite insurer is illustrative. Its figures are authored to teach the method and are not drawn from any firm.

The Mental Models

The Business Engineering Loop. Understand the distribution, abstract, compress, model, test against the real world, build intuition and return to the distribution. *Applied here:* the canvas is that loop run on a single process, and the standard and exception queue keep it running.

Know Your Distribution. Normal domains reward rules and averages; power-law domains are decided by the tail. *Applied here:* agents run the body of the claims distribution, and junctions sit in the tail where the money is.

Compression. State a mechanism in the fewest words that still predict something. *Applied here:* the canvas compresses a process manual into twelve questions, and the compression line compresses the method into three sentences.

The Executor Shift. Each generation designs work for the executor it has. *Applied here:* a century of methods assumed an executor that follows instructions; agents decide, so the design has to change.

Outside-In, Inside-Out. The web changed how firms meet their markets; AI has to change how they work inside. *Applied here:* the business model mapped the outside, and the core needs a map of its own.

VTDF. A business reads through four models: value, technology, distribution and financial. *Applied here:* each work loop's outcome traces to the Value model, and its economics roll into the Financial model.

Compensation Stripping. Many steps in a mature process exist only to work around constraints that no longer apply. *Applied here:* strike them before writing a rule, or the agent automates the workaround.

The Work Loop as Unit. Design around outcomes owned end to end, not tasks in a sequence. *Applied here:* one canvas per loop, never one per company or one per task.

The Four-Class Decision Sort. Every decision is mechanical, bounded, consequential or contested. *Applied here:* the sort decides what an agent may do alone and what must reach a person.

The Contested Decision. Some decisions were never agreed, and no engineering can settle them. *Applied here:* in the worked run, a goodwill policy held the loop at Shadow until a committee settled it.

The Junction. A point where a person decides, with defined evidence and a way to reverse the action. *Applied here:* Toyota′s stop cord, rebuilt for an executor that decides.

The Signed Node. Readiness counts only what is written, agreed and owned. *Applied here:* the gates read signatures, not confidence.

The Organizational Connectome. A firm is a network of loops joined by handoffs. *Applied here:* the Work Loop Map draws that wiring, so dependent loops scale together.

Acceptance Over Activity. Measure accepted outcomes, not tasks performed. *Applied here:* the meters report acceptance rate and cost per accepted outcome, never agents deployed.

The Thread: Shared Instruments Across the Library

The Work Loop Canvas is self-contained, and it also connects to the rest of the Library through instruments the volumes share.

The business engineering loop and **VTDF** come from *The Business Engineer*, the Foundation Volume, along with the Thinking OS behind them. This book runs the loop on a single process and adds the level beneath VTDF: VTDF says what the business is, and the work loops say how it runs.

The junction is the shared name for the point where a person decides. *The Business Orchestrator* treats it as the central design object of an AI-era firm. Here it becomes a node with a required reason, evidence and a way to reverse.

The charter from *The Redesign* is the full set of artifacts a process redesign produces, one per pass. The Work Loop Canvas is the one-page form of that charter. Teams that need the full depth of a redesign should run *The Redesign*'s eight passes and record the result on the canvas.

The standard is the subject of *The Standard*, which treats acceptance commercially, and *Evaluation Engineering*, which treats it technically. The standard node is where those volumes meet a single loop.

The property line from *The Enterprise Capture Test* asks what the firm keeps if a vendor walks away. In canvas terms, the rules, the decision sort, the standard and the junctions are the firm's property. They should never live only inside a vendor's product.

Also in This Series

This book is part of a series on embedding AI into the enterprise core. Each volume stands alone, and together they form one discipline.

The Business Engineering Body of Knowledge. Volume I, the foundation: from the laws of thought to closed loops of action, the core method, every concept and measure, and a library of mental models by type of loop.

The Work Loop Canvas. Volume II, this book: the instrument, one loop on one page.

The Embedding Lifecycle. Volume III, the sequence: ten signed steps from reading the map to running in the core.

The Work Loop Library. Volume IV, the record: thirty-nine reference loops across six chains, six filled in end to end.

The New Business Engineer. The series brought together into one book.

Also in this Library

The Business Engineer: the Foundation Volume, with the VTDF framework and the method behind this book.

The Redesign: the Process Volume, with the eight passes that take a documented process apart and rebuild it for human-and-agent execution.

The FDE Playbook: the forward-deployed engineer's field manual, whose first duty is mapping the work before embedding AI in it.

The Business Orchestrator: the Operator's Volume, on running a firm where judgment is the binding constraint.

The Agentic Organization: on how the firm and its careers change around AI.

The Standard: what enterprise AI has to prove, and who decides.

Evaluation Engineering: the technical discipline of grading what AI produces.

THE BUSINESS ENGINEER

The Work Loop Canvas

Strip the workarounds. Sort the decisions. Sign the loop.

businessengineer.ai