SAMPLE

Building the Automation of Tomorrow

Building the Self-Operating Systems of Tomorrow

Aiman JoeHanz

Free sample: Introduction & Chapter 1

putrabooks.github.io

Buy the full book:

Carousell carousell.sg/p/building-the-automation-of-tomorrow-ebook-pdf-1465868461

ElevenReader

Ko-fi

Payhip

Contents

Introduction

Chapter 1: The Definition

Chapter 2: The Anatomy

Chapter 3: The Senses

Chapter 4: The Controller

Chapter 5: The Actuators

Chapter 6: The Exception

Chapter 7: The Line

Chapter 8: The Program

Chapter 9: The Human

Chapter 10: The Operating Program

About this Series

Connect

This free sample contains the Introduction and Chapter 1. The complete book contains the full contents above.

Introduction

Automation is the oldest promise of the machine age and the least understood: a machine doing the work, but also deciding the work, without a hand on the lever at every step. This volume treats automation as what it has actually become — an engineering discipline with a definition, an anatomy, a failure catalog, and an operating discipline a builder can run — rather than as the vague future it is marketed as.

The definition this volume holds is strict, and the chapters enforce it. A system is automated when it executes a procedure without human intervention at each step, closing its own loop: it senses the state of the work, decides the next action against a model, acts on the work, and senses the result again. Mechanization, which multiplies the hand, is not automation; autonomy, which writes its own procedures, is beyond it. Between those two lies the discipline this book is about — the delegated procedure, executed faithfully, at a speed and a consistency no hand can hold, under supervision the design must make honest.

The volume's ten chapters take that discipline apart and put it back together. The definition and the contract come first, with the impostors the word keeps attracting. The anatomy follows — the senses, the state model, the controller, the actuators, and the supervision interface, the five organs every automated system shares and the one loop they all serve. Then the organs in depth: the sensing chapter on measurement honesty and calibration; the control chapter on the closed loop, its stability, and the tuning discipline; the actuation chapter on motion, power, wear, and the interlocks that keep both operators and machines whole; the exception chapter on alarms, degraded modes, and the paradox that the better the automation, the less practiced the human who must catch its failures.

The integration chapter builds from machine to line to plant; the software chapter takes on the program itself — the state explosion, the simulation before deployment, the versioning that keeps a running system repairable; the human chapter prices the supervision honestly; and the operating program closes the volume with the only questions that matter in a boardroom or a shop: what to automate, what it really costs, how it fails, and how it is kept.

The series promises hold as they have for thirty volumes: no vendor names, no benchmark statistics, no forecasts dressed as facts. Structure survives the product cycle; the particular factory that proves a principle this year retires it the next, but the principle — the loop, the contract, the exception path, the tuned controller, the practiced operator — holds across centuries of looms,

lines, and code. The reader who finishes this volume will be able to read any automated system, from a thermostat to a bottling line to a lights-out factory, as one structure: a procedure, delegated honestly, sensed truly, decided by a model the builders can explain, acted with power that respects its limits, and supervised by a human the design kept competent.

That is the whole of automation, said in one sentence — and the ten chapters that follow are that sentence, taken apart, so it can be built again.

Chapter 1: The Definition

Automation is the most used and least defined word in the industrial vocabulary. It is attached to a door that opens by itself, to a factory that runs without lights, to a script that moves a file, to a robot arm that welds a car, and to a software agent that writes its own emails — five systems with almost nothing in common except the word. Before anything can be built, the word has to be made small enough to hold, and this volume's first act is to make it small.

The definition this series builds on is one sentence: an automated system is a machine executing a procedure without human intervention at each step. Every word in that sentence is doing structural work, and the rest of the chapter takes it apart the way the rest of the book takes automation apart. A machine — meaning any engineered system, mechanical, electrical, or software. Executing — meaning actually doing, not recommending or describing. A procedure — meaning a defined sequence of actions, designed in advance by a human being who understood the work. Without human intervention at each step — meaning the human is present at the ends of the procedure, and at its exceptions, but not inside its loop.

That last clause is the one the market keeps quietly deleting, and the one this volume keeps. The difference between a power tool and an automated tool is not the power; it is the loop. A drill guided by a hand executes nothing by itself. A drilling cell that feeds itself parts, drills the hole, checks the depth, and rejects its own failures executes the procedure, and the human who set it up supervises the ends and the exceptions, not the steps. When the word automation is sold to mean anything less, the buyer has been sold mechanization, and the chapters that follow will show why confusing the two is expensive.

The definition stands between two neighbors that constantly borrow its clothes. One is mechanization — the machine that multiplies the hand. The loom, the press, the lift, the drill: mechanization gives the human power, speed, and precision, but the human remains in the loop of the work, deciding each step and initiating each action. Mechanization replaced the muscles of the worker; automation replaces the decisions of the operator. The transformation of the nineteenth century was mostly mechanization — machines that did the work while people ran the machines. The transformation that followed, and that this book is about, moved the running itself into the machine.

The other neighbor is autonomy — the machine that writes its own procedure.

A system that learns its own policy, sets its own goals, or generates its own sequence of actions is not executing a procedure that was designed; it is designing one, and the discipline that governs it is a different book, one this series has already given its own volume. The distinction matters because autonomy inherits every obligation of automation — the sensing, the acting, the supervision — and adds obligations automation does not have, and a builder who conflates the two will under-govern the autonomous system or over-govern the merely automated one. The volume's boundary is sharp and repeated: automation is delegated execution. What writes the delegation, and what audits it, is human work — and remains human work all the way through the coming decades of the technology.

Between the neighbors sits the discipline itself, and the middle position is not a compromise; it is a distinct engineering domain with its own mathematics, its own failure modes, and its own economics. The middle position is also where almost all of the value of the last century of industrial progress actually landed, which the coming chapters will keep demonstrating: the tuned loop, the faithful procedure, the honest exception path. Autonomy gets the keynote addresses. Automation runs the world's work.

Strip any automated system to its structure and the same four elements appear, arranged in a loop. The system senses the state of the work — the temperature of the batch, the position of the part, the presence of the file, the level of the tank. It decides, comparing what it senses against a model of what should be and selecting the next action from a designed policy. It acts, delivering the action to the work through whatever actuator the domain provides: a valve, a motor, a relay, a transaction. And it senses the result, which is where the loop closes and where the system becomes something a loop actually is: a structure whose next action depends on the outcome of its last one.

The open-loop system — the one that acts without sensing the result — is the boundary case that proves the definition. A toaster with a timer executes a procedure without sensing the bread; it is automated in the weakest sense, and its failure mode is familiar to anyone who has owned one: it executes the procedure perfectly on bread that needed something else. The closed loop is what the word should mean and, in serious systems, does mean — the difference between a machine that runs and a machine that runs well, between a procedure that executes and a procedure that achieves. The chapters that follow keep returning to this distinction, because the entire discipline of control, the subject of Chapter 4, exists to make the closed loop behave.

The loop also explains the one thing most explanations of automation miss: why automation fails differently from mechanization. A mechanized system fails the way machines always fail — a part breaks, a power cut stops the line, a tool wears out. An automated system can fail while running perfectly, because it can execute its procedure faithfully against a world the procedure no longer fits. The part is present, but not where the model expects; the material is harder than the model assumes; the sequence completes and the work is wrong. This failure mode — the perfect execution of an unsuited procedure — is the signature failure of the discipline, and the volume gives it a chapter of its own, because the entire supervision architecture of the next nine chapters exists to catch it.

Delegating work to a machine is a contract, and the contract has three clauses, whether the delegation is formalized in a requirements document or left implicit in a purchase order. The first clause is scope: the procedure, exactly, in full — what states the system will encounter, what actions it will take in each, and where its boundaries sit. The second clause is the exception path: what the system does when the world presents a state outside its scope, which in honest designs means stopping safely and summoning the human, and in dishonest designs means whatever the untested code happens to do. The third clause is supervision: what the human operator can see, what the human operator can do, and what the system does when the human does it.

Every clause that is left unwritten at design time gets written at failure time, by the failure. The scope that was never bounded becomes the batch of material the system cheerfully ruins. The exception path that was never designed becomes the two o'clock phone call to the engineer who wrote the logic, and the answer is always the same: nobody ever tested that. The supervision that was never architected becomes the operator mashing the stop button while the system, following its own logic to the end, finishes ruining the work before the button's effect reaches the actuator — because the stop path was wired through the same controller that was busy.

The contract is the volume's spine. The anatomy chapters (2 through 5) build the parts that keep the first clause honest: sensing that tells the truth, decisions that are inspectable, action that respects its limits. The exception chapter (6) builds the second clause into the machine itself. The human chapter (9) prices the third. And the operating chapter (10) returns to the contract at the moment it matters commercially — the moment a buyer is standing in front of a proposal, deciding which clauses the vendor actually wrote, and which ones the buyer is about to discover at the worst possible hour.

Because automation is a word with budget attached, it attracts systems that are not it. The demo that runs a scripted sequence once, under supervision, on prepared inputs — automation is a property of the second thousand cycles, not the first one. The dashboard that displays a state and recommends an action

— that is decision support, a discipline with its own honest value and its own chapter of this series elsewhere, but the human remains in the loop at every step, and calling it automation confuses the audit: the system's failures will be blamed on the operator, and the operator's judgment will be credited to the system.

The efficiency project that removes steps from a manual procedure — process improvement is real and honorable, but it optimizes the human's work rather than delegating it. The machine-learning model that predicts the failure — prediction is sensing, done expensively and well, but a prediction that actuates nothing is a sensor with a publishing department. Each of these impostors can be part of an automated system, and in the best designs each one is: the prediction feeds the loop, the dashboard feeds the supervisor, the process improvement shapes the procedure before the procedure is delegated. The impostor is only an impostor when it takes the name of the whole, and the whole is the loop that closes.

The volume's test for the name is the volume's definition, and the reader can run it on any claim that crosses a desk: does the system execute a designed procedure, sensing the work, deciding the next action, and acting on the result, without a human inside the loop? If yes, the system is automated, and everything in this book applies to it. If no, the claim is selling a component as a machine, and the correct response is not rejection but reclassification — the component may be excellent, and it still needs the loop, the exception path, and the supervision before it is what the proposal says it is.

The definition reads as though it fell from the sky, and the chapter owes the reader its history, because the history explains why the definition is shaped the way it is. The first delegation of execution was mechanical: the governor, the float, the escapement — machines that held a quantity to a target without a hand on the valve, and the first closed loops the industrial world ran. The second delegation was electromechanical: the relay panel, the timer drum, the hardwired sequence that ran a machine through its steps at machine speed, with its logic drawn — literally — as the circuit that implemented it.

The third was electronic: the programmable controller, born when the relay panels grew past the size a human could rewire, and the logic moved from copper into memory while keeping the circuit language its maintainers could read. The fourth is the one the reader lives in: the logic is software, the

machine is a system, the line is a network, and the procedure is written in the same media as everything else the organization runs.

The history's structural lesson is that each delegation kept the previous one's shape. The governor's loop is the PID's loop; the relay panel's interlock is the ladder rung that survives in every industrial controller on earth; the drum timer's sequence is the state machine wearing a different suit. The reader who learns the loop, the interlock, and the sequence has learned the transistive core of four generations of the technology, and the marketing that announces each new generation as a rupture is describing the clothes, not the body. The definition this chapter built — delegated execution of a designed procedure, in a closed loop, with a supervised human at the ends — has been true of every generation, and the volume's bet, argued across its ten chapters, is that it will be true of the next one too.

The history also explains the discipline's most persistent confusion, the one the chapter opened with: why the word automation keeps being attached to things that are not it. Each generation arrived with a wave of systems that did less than the word claimed — the mechanization that was sold as automation, the assist that was sold as delegation, the dashboard that was sold as a machine — because the word was the budget line and the budget line was the point. The definition's strictness is not pedantry; it is the audit that keeps the budget honest, and the reader who applies it will find, in any era's catalog, that the systems which pass it are the systems the next fifty years of maintenance schedules are written about.

Between the fully manual procedure and the fully automated one lies a ladder of delegation, and the ladder deserves its own section because most real systems stand on its middle rungs. The first level is assistance: the machine does a part of each step — the power tool, the autocomplete, the guided picking — and the human remains inside the loop of every cycle. The second level is partial automation: the machine runs some steps of the procedure without the human and hands the others back — the machine loads itself and the human unloads it, the script deploys and the human approves. The third level is the full loop: the machine executes the whole procedure, and the human supervises from outside the cycle, at the ends and the exceptions. The fourth level is the exception-only human: the machine runs the procedure, handles its own absorbable deviations, and the human's entire job is the failure that must escalate.

The ladder is not a maturity scale, and the chapter says so against a decade of consulting decks: the higher rungs are not better, they are more delegated, and delegation is a trade whose terms the previous sections set — scope,

exception paths, supervision, practice, and cost. Some procedures belong on the first rung forever, because the judgment inside each step is the value, and the machine's assistance is the whole honest claim. Some belong on the fourth, because the procedure is understood, frequent, and drudgerous, and the human's competence is best spent on the failures.

The design sin is not standing on a low rung; it is standing on a rung the system pretends is higher — the third-rung system with a first-rung human pacing it, or the fourth-rung claim wrapped around a procedure nobody has ever fully written down. The buyer's honest question is not how automated the system is, but which rung it truly stands on, and whether the rung below it — the fallback — has been built and drilled, because the ladder is descended at three in the morning, in the dark, at speed.

The ladder's engineering content is the handoffs. Each rung boundary is a place where the work passes between the human and the machine, and each handoff is a designed interface with its own failure modes: the handoff that loses the state (the machine was on step four, and the human took over at a moment nobody recorded), the handoff that loses the intent (the human takes over the machine's half-finished action without the machine's reason for starting it), and the handoff that loses the authority (the machine continues acting on its own model while the human believes they have control). The volume's later chapters build the machinery — the modes, the displays, the interlocks — that make the handoffs honest, and the reader should carry the ladder forward as the map of where those machines matter most.

The chapter's last depth is a practical one: the audit a reader can run in one meeting, against any automation claim that crosses their desk. The claim is presented as a deck: the system, its benefits, its integration, its price. The audit is the definition, asked as five questions. What is the procedure, written down? — and the deck that cannot produce the procedure is selling a capability, not a system, because the scope clause of the contract is the procedure, and a system without one has an unbounded scope and no exception logic to catch what the scope missed.

What closes the loop? — the claim's mechanism for sensing its own results, because the loopless system is a toaster, and its failures will be discovered by whoever consumes its output. What is the exception path? — the state the system enters when the world surprises it, the boundedness of its retries, and the safe state's wiring; the deck that answers "the operators handle that" has delegated the design to the crew, and the crew has not been consulted. Where is the human, and what keeps them competent? — the supervision's display, the alarm's philosophy, and the manual path's drills, because the definition's

last clause is the system's most expensive to rediscover.

And what is the ledger? — the whole cost, the program that keeps the truth, and the exit, because the claim that has no retirement has not finished being designed.

The five questions take ten minutes, and they work because they are the definition's, not the auditor's: each question is one clause of the contract, and the deck that answers all five honestly has done the engineering the definition demands, whatever its technology. The deck that fails the five has not failed an interrogation; it has shown, before the purchase order, which unwritten clause its failures will be found writing later. The reader who carries the five questions carries the chapter, and the reader who carries the chapter carries the definition into every room the word automation is said in — which is the volume's entire purpose, stated at its beginning.

The definition has critics, and the chapter is stronger for meeting them. The first says the definition is too strict — that the door that opens by itself, the script that moves the file, the assistant that drafts the reply are automation to the people who benefit from them, and the discipline that denies the name is gatekeeping. The answer is the taxonomy, not the veto: each of those systems is real and useful, and each is a rung of the delegation ladder — assistance, partial automation, the loopless sequence — and the name matters only because the obligations differ. The door owes a sensor and a timer; the script owes an error path; the assistant owes the review its content requires. The system that is called automation owes the whole contract — scope, exception, supervision — and the critic's own complaint dissolves when the rung is named, because the honest rung has honest obligations, and the wrong name is what sold the wrong obligations to the wrong buyer.

The second critic says the definition is too loose — that the machine that executes a procedure but cannot adapt has no right to the word, which the coming decades reserve for systems that learn. The answer is the volume's boundary, restated as a compliment: automation is the delegated procedure, and the learning system is the procedure's author's apprentice, a different discipline with its own volume, and the mixing of the two is precisely what the strict definition prevents. The discipline that keeps automation's name for the executable, bounded, supervised procedure is the discipline that keeps the audit possible — and the audit, the whole volume has argued, is what the word is for.

The clause that separates automation from its neighbors — without human intervention at each step — contains the entire human story of the discipline, and the volume closes its first chapter by holding that clause up to the light.

The human remains, at the ends of the procedure and at its exceptions. The design of the system determines whether that human is a supervisor with the competence, the visibility, and the authority to catch the system's failures — or a liability, blamed for a failure the design guaranteed and equipped no one to prevent.

The distinction is not sentimental; it is structural. The automated system that keeps its human competent is designed differently from the one that does not: its alarms are honest (Chapter 6), its displays show the state the machine believes, not the state a vendor wanted to show (Chapter 9), its procedures keep the human practiced at the manual path (Chapter 10), and its failures are recorded as design information rather than disciplinary files. The system that does not do these things fails at its exceptions — and the failure catalog of the discipline is, almost without exception, a catalog of systems that treated the last clause as a headcount reduction instead of a role redefinition.

The chapter's summary, held in one breath: automation is delegated execution of a designed procedure; the delegation has scope, exception paths, and supervision as its contract; the loop that closes — sense, decide, act, sense again — is the structure that makes the delegation real; and the human at the ends of the procedure is not the residue of the old way of working but the load-bearing element of the new one. The chapters that follow build each element in turn, and the reader who holds the definition will find that every one of them is a way of keeping it honest.

End of free sample. The full book is available at putrabooks.github.io

ABOUT THE AUTHOR

Aiman JoeHanz writes the Building of Tomorrow series, a volume-per-technology account of the systems defining this decade -- each one a structural treatment of the parts, the failure modes, the economics, and the operating discipline a builder can run. In this volume the author takes the series to the delegation itself: automation, the machine that executes the work and decides the next step -- treated here as the strict definition and its impostors, the five-organ anatomy and its seams, the honest senses and their calibration program, the closed loop and its tuning discipline, the force of the actuators and the safety machinery that respects it, the exception ladder and the supervision paradox, the line and its weakest machine, the program as an artifact, the human as the system's last organ, and the operating program that keeps the whole truth on a calendar. The catalog applies the same lens to everything else the decade is building.