

RTSAX – Real-Time Society, Architecture and Execution

Architecture for the Human–Synthetic Enterprise

Booklet · Version 2.0

Otto Th.G. van Haaren



RTSAX – Real-Time Society, Architecture and Execution

Architecture for the Human–Synthetic Enterprise

Booklet · Version 2.0

Otto Th.G. van Haaren

2026

Colophon

Authorship and Rights:

All intellectual property rights related to this document are held exclusively by Otto Th. G. van Haaren. The content has been created in a personal capacity and solely represents the author's vision, reflection, and ideas.

Any references to, or insights derived from, previous work experience have been incorporated based on personal development and knowledge building, and in no way give rise to collective ownership, co-authorship, or intellectual property claims by third parties (such as organizations or former employers). This work is protected by copyright; reuse or distribution is permitted only with the express written consent of the author. Excerpts may be quoted only with proper source attribution and acknowledgment of the exclusive copyright.

Version 2.0, October 2026. Version 1.0 appeared in September 2025. This version was co-written and edited with two AI models, Claude (Anthropic) and ChatGPT (OpenAI), under the author's direction; the vocabulary and the decisions of this version are proposed by the author and remain his to ratify.

Preface

One Year On

When I founded 5CA in 1998, I wanted to understand how technology could help us organise work better. Nearly thirty years later, that curiosity remains.

Our first AI engineers joined in 2018 to help us understand AI and explore its applications. Our core business is customer support. We apply technology to help our clients serve their customers better.

A year after this book's first edition, AI systems are increasingly able to choose and execute actions independently. A new participant is emerging within the enterprise: a synthetic actor.

As a leader of 5CA, I must understand what this means for our organisation and our responsibilities towards employees and clients. That is a central motivation for this book.

The challenge is increasingly one of governance. Intelligence provides capability. It does not confer a mandate.

When AI acts on an enterprise's behalf, authority, boundaries and accountability must be explicit. Governance must operate during execution, while people retain the understanding and means to direct and intervene.

This is the premise of the Human-Synthetic Enterprise: people and synthetic actors working together within a shared governance framework.

I initiated this book and developed its thinking with David Bos, supported by ChatGPT and Claude in exploring ideas, testing arguments and drafting. Direction and responsibility remain with David and me.

This is a design hypothesis to be tested, centred on one question:

What kind of enterprise do we want to be when synthetic actors can also act on our behalf?

Otto Th.G. van Haaren

October 2026

Table of Contents

Preface.....	3
Table of Contents.....	4
Executive Summary.....	6
1 RTS – Real-Time Society (Why?).....	14
1.1 The need for reordering.....	14
1.2 Core principles: Rhythm, Norms, Semantics.....	15
1.3 AI as a synthetic actor.....	17
1.4 Humans in the driver’s seat: from hierarchy to semantic direction.....	19
1.5 The enterprise as governable space.....	20
2 RTA – Real-Time Architecture (How?).....	22
2.1 The design hypothesis: normative reasoning inside the runtime.....	22
2.2 The runtime core: four layers and the Runtime Control Loop.....	23
2.3 Eight layers of intelligent collaboration.....	26
2.4 Corpus, MetaCorpus and the container hierarchy.....	28
2.5 CoreSOS and IMUS.....	29
2.6 Mandate: who may do what, and how we know.....	31
2.7 Rhythm: Pulse and Pendula.....	33
2.8 Orchestration instead of loose agents.....	35
3 The Human–Synthetic Enterprise: seven deliverables (What must an organization put in place?).....	37
3.1 From technology to enterprise.....	37
3.2 Deliverable I – The Enterprise Constitution.....	39
3.3 Deliverable II – Mandate and Decision-Rights Architecture.....	41
3.4 Deliverable III – The Human–Synthetic Operating Model.....	43

3.5 Deliverable IV – Human–Synthetic Workforce Architecture.....	45
3.6 Deliverable V – Semantic and Context Architecture.....	47
3.7 Deliverable VI – Runtime Governance and Evidence Architecture.....	49
3.8 Deliverable VII – Learning and Constitutional Evolution Architecture.....	51
4 RTX – Real-Time Execution (What?).....	53
4.1 From model to practice.....	53
4.2 Application in various domains.....	54
4.3 Practical example: a refund request through the chain.....	56
4.4 From pilot to operating model.....	57
4.5 Conclusion: RTS, RTA and RTX together.....	59
Annex 1 – Glossary of terms.....	61
Annex 2 – Layered normativity: from universal to individual.....	65
The ten levels, from macro to micro.....	66
When levels conflict.....	68
Annex 3 – What has changed since version 1.0.....	69
Annex 4 – Literature.....	71
Chapter 1: RTS – Real-Time Society (Why?).....	71
Chapter 2: RTA – Real-Time Architecture (How?).....	72
Chapter 3: The Human–Synthetic Enterprise: seven deliverables (What must an organization put in place?).....	72
Chapter 4: RTX – Real-Time Execution (What?).....	73
Source documents.....	74

Executive Summary

AI is developing faster than organizations can absorb it. In recent years the emphasis has been on better models, more data, copilots, agents and automation. That development continues. But the main problem is shifting.

AI capability develops faster than adoption, and adoption faster than organization design. The limiting factor is therefore less and less the intelligence of the model, and more and more the enterprise's ability to let that intelligence act with meaning, with mandate and with effect.

As intelligence becomes more abundant, organizational scarcity shifts to meaning, mandate, execution and responsibility.

RTSAX is a design hypothesis for that shift.

From instrument to synthetic actor

Traditional software is mainly an instrument. Agentic AI can increasingly interpret context, weigh alternatives, choose next steps, use tools and initiate actions on its own.

RTSAX calls a system that is given such room to act a synthetic actor. That does not mean AI becomes human or conscious. It means the system can act on behalf of the enterprise.

This creates a fundamental distinction between what a system can do and what it may do.

An AI system can be technically perfectly capable of refunding 250 euros to a customer. That does not mean it has the mandate to do so. Perhaps the mandate is 100 euros, perhaps 300 euros under conditions, and perhaps human approval is always required.

Intelligence gives capability. It does not give mandate.

A more powerful model or a new tool can expand capability almost immediately. Mandate should change only when the enterprise deliberately decides so.

From intelligence to meaning and execution

Foundation models will become more powerful, and enterprises will increasingly use several models and vendors. The model itself therefore becomes less suitable as the organizing principle of the enterprise.

The more durable architecture lies around it.

The infrastructure connects intelligence to data, systems and execution. The ontology records how the enterprise understands its reality: what is a customer, an appointment, quality, risk, responsibility or an exception? Semantics ensures that human and machine mean sufficiently the same thing.

Otherwise an intelligent system can give a convincing answer and still be organizationally wrong.

But meaning is not enough either. A good analysis that is not executed creates little value. The relevant chain therefore becomes:

meaning → mandate → decision → execution → evidence → learning

RTSAX separates interpretation from authority. A system can understand perfectly what is happening without thereby acquiring the right to act.

The canonical formula is: *Intelligence proposes. Authority permits. Mandate bounds. Execution acts. Evidence proves.*

Governance moves to runtime

Traditional governance works mainly before and after the action. Beforehand, policies, procedures and powers are set. Afterwards come QA, compliance and audit.

That becomes insufficient when synthetic actors can act at scale and at machine speed.

Governance therefore has to move closer to the moment of action. RTSAX distinguishes four runtime functions:

- **State** – what is happening here?
- **Authority** – who or what may act?
- **Execution** – how is the authorized action carried out?
- **Accountability** – what happened, and can it be proven?

The outcome then becomes input again for perception, authority and execution. Governance becomes a closed control loop.

This does not mean a human has to approve every AI action. AI, too, can perform monitoring, verification and anomaly detection. But an actor should not at the same time set its own norm, execute it and pass the final judgment on it.

From implicit culture to explicit constitution

Much of what governs an enterprise is written down nowhere in full. It lives in culture, experience, relationships and shared expectations. People interpret that implicit reality continuously.

That cannot simply be expected of a synthetic actor.

The Enterprise Constitution therefore makes more explicit what the enterprise exists for, which fundamental principles apply, which limits may not be crossed, and which powers can and cannot be delegated.

The aim is not to lay down every situation in rules beforehand. A good constitution should make responsible independent action possible.

Beneath it, three functions continuously interact: rhythm, norms and semantics. Rhythm determines when an enterprise acts, reflects and corrects. Norms connect ethical principles to limits on action. Semantics provides shared meaning.

Speed also requires slowness

A real-time enterprise is not an enterprise in which everything happens as fast as possible.

Faster systems can also overreact faster. Extreme efficiency can remove the buffers needed to absorb unexpected situations.

That is why RTSAX introduces Pulse and Pendula.

The Pulse is the agreed cadence at which the enterprise re-assesses its current state. Pendula are deliberate counter-movements meant to prevent overreaction, drift and fragility.

The aim, then, is not maximum speed but responsible speed.

The organization itself changes

The classic enterprise is largely designed around human scarcity. People hold knowledge, carry out work and take decisions. Management layers coordinate those people, that information and those powers.

When synthetic actors take over more and more perception, analysis and execution, that logic changes.

The organization chart does not disappear, but it becomes insufficient to describe how the enterprise really functions.

Alongside the hierarchy a mandate network emerges: human and synthetic actors, workflows, decision rights, responsibilities and escalation paths. The management question thus shifts from only “who reports to whom?” to “who or what holds which mandate and which responsibility, within which domain?”

Human value shifts, relatively, to judgment under uncertainty, exceptions, meaning-making, workflow design, normative conflicts, mandate and responsibility.

Seven design responsibilities

RTSAX translates this into seven things a Human–Synthetic Enterprise has to design:

1. **Enterprise Constitution** – purpose, principles, fundamental limits and reserved powers.
2. **Mandate and Decision-Rights Architecture** – who or what may decide and act on what?
3. **Human–Synthetic Operating Model** – how do work, workflows, exceptions and human–AI collaboration move through the organization?
4. **Human–Synthetic Workforce Architecture** – which work is done by humans, augmented humans and synthetic actors?

5. **Semantic and Context Architecture** – how does the enterprise organize ontology, knowledge, context and shared meaning?
6. **Runtime Governance and Evidence Architecture** – how do authorization, intervention, control and evidence become part of execution?
7. **Learning and Constitutional Evolution Architecture** – how can the enterprise learn without systems silently rewriting their own mandate or constitutional limits?

These are not seven software products. They are seven enterprise design responsibilities. Technology platforms can supply more and more of the technical infrastructure, but the enterprise remains responsible for its meaning, its design and its limits.

Learning without losing control

A real-time enterprise cannot be static. Technology, legislation, markets and expectations change continuously.

So the architecture, too, must be able to learn.

But learning must not mean that a system imperceptibly extends its own powers. Operational patterns can be adjusted quickly; fundamental changes to purpose, rights, ethical limits or reserved powers require heavier human decision-making (Bounded Constitutional Evolution).

The rule is: *the enterprise must be able to learn without imperceptibly becoming a different enterprise.*

Measure first, then scale

RTSAX is not a proven universal organization model. It is a testable design hypothesis.

That is why measurement is essential. Not only productivity, cost and automation rates, but also exceptions, human overrides, mandate breaches, decision quality, drift, evidence and realized outcomes.

The practical rule is therefore: *measure first, then design deliberately, and only then scale autonomy.*

Start small, for example with one concrete decision moment such as a refund. Make explicit which actor may decide, which mandate applies, when escalation is needed, which evidence is recorded and what is learned from the outcome.

5CA as design context

In this booklet 5CA is a design context, not the proof that RTSAX works as a whole.

5CA's role is not to build foundation models. Technology companies will go on developing models, cloud infrastructure, and data, identity, security and agent platforms.

Its challenge lies closer to execution: configuring, connecting and organizing that technology so that human and synthetic actors can act reliably within concrete customer-experience processes.

AICX, 5CA's AI Customer Experience Corpus, is the first execution context for this. The broader question of RTSAX then is: which context does an actor need, which mandate applies, when may the system act, when must it stop, where does human judgment remain necessary, and which evidence must remain?

The human question

The aim, ultimately, is not maximum automation.

When AI frees up capacity, the question arises what we use that capacity for. More efficiency can easily lead to more production and, again, more work.

The Human–Synthetic Enterprise must therefore determine where synthetic execution creates value, and where human judgment, responsibility, relationships, creativity, craftsmanship and meaning must remain central.

The aim is not to remove the human from the system. Nor is it to keep the human artificially in every process.

The aim is the right division of agency.

With that, RTSAX returns to one central question:

How do we organize meaning, mandate, execution and responsibility when human and synthetic actors act on behalf of the same enterprise?

RTSAX offers no final model for that, but a design hypothesis that has to be tested in practice.

1 RTS – Real-Time Society (Why?)

1.1 The need for reordering

At first glance, much of today's use of AI still looks harmless: summarizing a text, rephrasing an email, generating an image. Yet AI is embedding itself ever deeper in the foundations of our language, our choices and our thinking.

This raises a fundamental question: who safeguards meaning when machines increasingly make the first move? Meaning is not neutral output. It is a field of tension in which values collide and context matters. If AI takes over that space through pre-selections, filters and summaries, the power over meaning shifts from human to system – not with a bang, but in small, comfortable steps.

The answer begins with semantic sovereignty: societies, organizations and individuals must remain the ones who define what their words and their norms mean. Two examples:

- **In AI systems** – An algorithm that decides who is “eligible” for a mortgage or a job implicitly defines what eligibility means. An organization that has not defined the term itself leaves that definition to the system.
- **In public decision-making** – A government that bases policy on external AI services such as fraud detection loses semantic sovereignty if it does not itself control the definitions of “risk” and “fraud.”

In 2026 the question has become sharper, because AI no longer only advises: it acts. AI agents execute tasks, take decisions within the bounds

they are given and learn from the results. The machine now makes many of the next moves as well.

Meanwhile three developments move at different speeds: the capability of AI grows exponentially, its adoption inside organizations linearly at best, and the design of organizations – jobs, decision rights, governance – at a glacial pace.

The visible result is the pilot trap. A team builds a demonstration that looks good in the room. Nothing changes in daily operations, because the workflow, the data and the role of the human reviewer stay as they were. The organization collects pilots and concludes that AI does not work.

The constraint is therefore organizational, not technical. Most organizations run on implicit coherence: experienced people know how things really work and repair inconsistencies informally. An agent cannot do that. The smarter the system becomes, the more visible the incoherence around it. The book sets this out in detail (Chapter 1).

What is needed is not a better model but a reordering of how decisions are made, how meaning is preserved and how rhythm is kept. The Real-Time Society (RTS) is the vision behind that reordering: a search for a balance in which humans and AI evolve together on the basis of shared meaning.

1.2 Core principles: Rhythm, Norms, Semantics

RTS is built on three interrelated principles – the RNS triad. They are the three load-bearing functions of any stable order, whether a society, an organization or a technical system, and AI puts all three under pressure at once.

- **Rhythm** – The cadence at which a system observes, decides, acts and learns; speed is not rhythm. An agent working at machine speed can revise a decision a thousand times between two weekly reviews, so rhythm can no longer be left to habit.
- **Norms** – The agreements about what is permitted, required and forbidden; they give direction and set boundaries. An agent has no moral compass of its own, so norms must be present at the moment of action, not only in rules beforehand or in liability afterwards.
- **Semantics** – The shared meaning that lets people and systems understand the same words in the same way. For example: in a coherent organization “customer” means the same thing in sales, support and finance. An agent does not merely use language; it converts language into action.

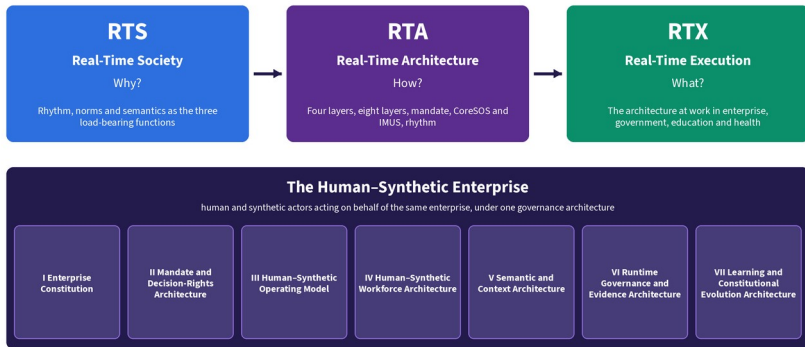
Version 1.0 of this booklet called this the RES triangle: Rhythm, Ethics, Semantics. Ethics has not disappeared. It is the tradition from which norms are drawn, and norms are the form in which ethics enters a system: ethics lives within norms.

Of the three, rhythm is the most neglected: most approaches treat the governance of AI as a fixed set of rules to be checked, whereas RTS treats it as a living cadence that has to be designed.

In practice the triad comes down to three questions that an AI system should be able to answer at the moment of decision. Is this the right time and pace? Can this action be justified within the norms that apply? Do all parties, human and machine, mean the same thing by the words they use?

The figure shows the whole, RTSAX: RTS as the Why, Real-Time Architecture (RTA) as the How and Real-Time Execution (RTX) as the What,

together with the Human–Synthetic Enterprise and its seven deliverables (section 3.1).



RTSAX: Real-Time Society (Why), Real-Time Architecture (How) and Real-Time Execution (What), and the Human–Synthetic Enterprise with its seven deliverables

1.3 AI as a synthetic actor

Every earlier technology was an instrument: it did what it was told and stopped when told to stop, and the initiative stayed with the human user. Today's AI is different. A model interprets an incomplete instruction and produces an output the user did not write; an agent that is given a goal starts a chain of actions that nobody approved one by one.

RTS therefore treats AI as a synthetic actor: a non-human entity that observes, interprets and acts within human systems, with consequences for which people remain accountable. Four characteristics set it apart from an instrument: it works without step-by-step instruction, acts on interpreted meaning, learns from feedback and influences others beyond its own task. This is a statement about governance, not about consciousness.

Agency is assigned, not assumed. In the governance sense, a system does not become an agent because it reasons well, but because an organization gives it room to act – and can take that room back. What makes a system an agent is not its intelligence; it is its mandate.

Two configurations follow. In Tool mode the goal is set from outside, the task is bounded and nothing takes effect without human approval, given per case or in advance for a fixed pattern. It is the default, and often a deliberate end state. Actor mode gives the system discretion within explicit, revocable bounds, and only after an explicit decision for which a named human, the Actor Sponsor, is responsible.

For example: a system that drafts a reply for an employee to approve works in Tool mode; one that handles customer requests on its own, choosing among permitted paths within hard limits, works in Actor mode. The book sets this out in detail (Chapters 5 and 13).

The most consequential mistake is to treat an agent as a digital employee: to give it a persona, to “onboard” and “coach” it, and to forgive its errors as “learning.” An agent is managed without humanizing it: it is specified, bounded, instrumented and audited. The interface may smile. The mandate must not.

Synthetic actors carry functional accountability only: what they do is attributable, traceable, bounded and verifiable. Responsibility stays with the humans who grant, bound and revoke the mandate.

Finally, synthetic actors add capacity, and supply creates demand. It is Parkinson’s law: work expands to fill the time available. Freed capacity is re-absorbed – through more requests, more elaborate work and more coordination – unless it is deliberately protected. Otherwise AI tends to deliver more activity at higher speed, not more time for what matters.

1.4 Humans in the driver's seat: from hierarchy to semantic direction

AI makes access to knowledge, analysis and first interpretation drastically cheaper. Knowledge has become a commodity. Meaning is the new scarce capability. Meaning here is not vague inspiration. It has three functions: direction (why is this being done?), coherence (how does it fit the larger whole?) and mattering (why is this choice valuable?).

For example: a customer asks for a refund. The request contains no answer of its own. Before anyone can decide, someone has to give meaning to what is happening. Meaning comes before the decision, and the decision before the execution.

The classical organization is built on hierarchy, which answers three questions: who decides, who executes and who controls. An agent can do all three within a single decision cycle. The human role does not disappear. It shifts from positional authority to semantic direction: the responsibility for shaping, defending and recalibrating the meaning within which the system operates.

Hierarchy does not vanish, but it becomes a line of meaning rather than a line of command. The periodic question “did we hit our targets?” makes way for a continuous one: “are we still acting in line with what we meant?”

The manager's role changes most, because the object of management moves from the team to the system of people and AI together. The manager becomes a semantic authority: not the owner of the truth, but the person who determines what counts, which interpretation leads, which norm applies and where a mandate begins and ends.

Meaning First turns this into a design constraint rather than a slogan: where AI absorbs routine cognitive work, performance depends less on producing output and more on governing meaning. Four constraints make it practical: every role has an explicit decision space; AI may execute, but may not decide normatively or expand its own mandate; meaning is operationally defined; and governance works through rhythm rather than command. The book sets this out in detail (Chapter 6).

What AI frees is time and capacity. What it does not free is meaning. Creating space is technical, and solvable; using it for judgment and direction rather than for more of the same is cultural, and AI cannot do that for us. The risk is not that AI becomes too smart, but that we stop practicing the act of giving meaning. Humans in the driver's seat therefore means: not above the system as commander and not below it as executor, but inside it, giving direction and safeguarding meaning.

1.5 The enterprise as governable space

RTS is a lens on society, not a blueprint for it. A society cannot be governed in real time: it is an open system that nobody controls and nobody can design. Visions that expect technology to make the world governable do not overestimate technology; they underestimate complexity.

An enterprise is different. It is a bounded space of explicit relations – contracts, roles, workflows and shared objectives – that people enter voluntarily. That does not make it controllable, but it can organize what society finds ever harder to organize: direction, explicit meaning, clear normative limits and a consciously chosen rhythm. It can grant explicit mandates, and it can lay down in one constitution what it stands for, what it does not do and which limits apply.

Governable does not mean controlled. Control presupposes predictability, complete information and central mastery. Governability is the property of a system that accepts uncertainty and deviation while keeping direction, boundaries, rhythm and correction intact. Once agents act at a scale and speed that classical management cannot follow, full central control is impossible, and governance has to move into the running system itself. Governance of this kind is not a brake on autonomy; it is its condition.

For example: an agent that prepares refunds can be technically correct and culturally wrong, because it inherits no company culture and knows no unwritten rules. What people have learned over the years must therefore be made explicit in principles, definitions, mandates and limits.

Rhythm belongs here too. It is not a well-being theme but a governance variable: a system has rhythm when acceleration, reflection, correction and stabilization stay in balance. An organization that confuses speed with adaptivity becomes technologically stronger and humanly more fragile.

That is why the architecture in this booklet is built at the level of the enterprise. RTS leaves the world as it is; RTA makes the enterprise governable within that world. Once AI sets priorities or assesses performance, a governance layer exists inside the enterprise whether or not anyone has designed it. The next chapter shows how to design it.

2 RTA – Real-Time Architecture (How?)

2.1 The design hypothesis: normative reasoning inside the runtime

To bring the RTS vision into practice, Real-Time Architecture (RTA) starts from one design hypothesis. As soon as systems act in normatively charged contexts, at machine speed and at scale, the normative reasoning that governs them must move into the runtime architecture – into the system as it operates. It can no longer remain external policy, reviewed after the fact.

A context is normatively charged when the question “did the system do the right thing?” depends on more than functional correctness. For example: a hiring shortlist can follow every criterion and still discriminate. In such contexts the norms are made explicit inside the system, interpreted at the moment of action, and weighed within mandates set by accountable humans.

The hypothesis does not claim that AI becomes morally sovereign or generates its own values. The norms are written by humans; the system applies them to specific cases, much as legal reasoning does. Human responsibility is not reduced (section 1.3).

Where should normative reasoning live? The market shows three positions:

- **AI as a tool** – AI is an instrument inside human processes, and all normative judgment stays with people. This holds only while a human reviews every decision; at the scale of agents, the volume defeats it.

- **Externalized normativity** – AI acts, but the norms sit outside it: policy encoded as guardrails at the boundary, with escalation and audit afterward. On a defensible reading, the major platform vendors have broadly taken this position, and rationally so: they serve thousands of clients under hundreds of regulatory regimes.
- **In-system normativity** – The normative reasoning is itself part of the runtime. This is the position of RTA. It fits organizations that are operationally close to their own decisions: they see the consequences within days and have the authority to revise their own norms.

The choice is positional, not ideological. A platform vendor cannot carry the norms of every client; an enterprise that leaves all its norms to the platform gives up a governance asset that only it can use.

The hypothesis responds to one structural error: mistaking permission for authority. Many systems act because access rights and workflow settings allow it, not because anyone made the action legitimate. A permission determines access; a mandate determines what may legitimately be done. In short: “can” is not “may.” The book sets this out in detail (Chapter 8).

2.2 The runtime core: four layers and the Runtime Control Loop

The hypothesis becomes architecture in four functional layers. Every action – by a human or by a synthetic actor – passes through all four, so that questions that organizations usually answer implicitly are answered explicitly, at the moment of action:

- **RTA1 – Real-Time State Architecture** – What is happening, in what context and with what relevant information? Data is not state: state is the current, decision-ready picture built from scattered data.

- **RTA2 – Real-Time Authority Architecture** – What is permitted to happen, by whom and within what limits? Legitimacy is applied at the decision gate, which every action must pass.
- **RTA3 – Real-Time Execution Architecture** – What actually happens? This layer executes, in bounded steps, what RTA2 has permitted. It does not decide.
- **RTA4 – Real-Time Accountability Architecture** – What can be shown to have happened? This layer provides evidence and traceability for every action.

Together the four layers form the runtime core, a closed control loop: what RTA4 establishes about one action returns to RTA1 as part of the state for the next. Each layer is necessary and none is sufficient. Execution without authority, for instance, is efficient but illegitimate.

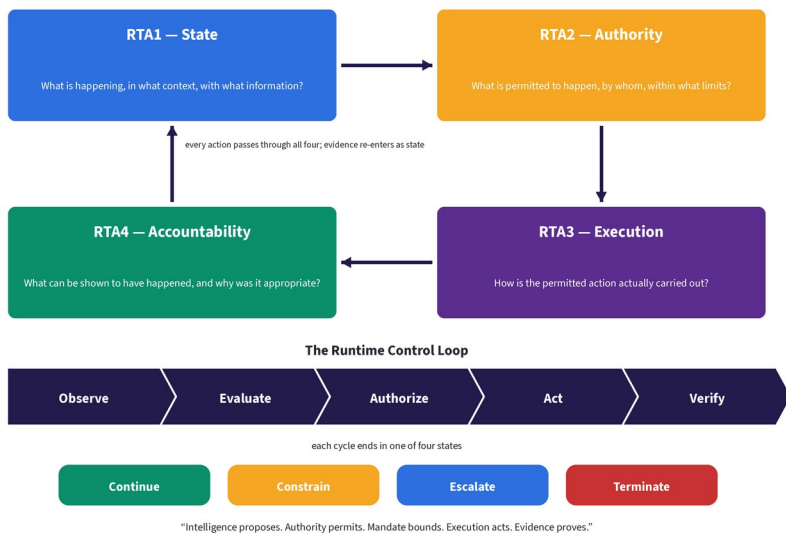
The Runtime Control Loop is the governance cycle that runs across the layers. It turns a changing reality into a controlled action in five steps:

1. Observe – establish the relevant state (RTA1).
2. Evaluate – interpret the situation and develop options; none of them is yet permitted.
3. Authorize – test the proposed action against authority, mandate and current context (RTA2).
4. Act – carry out the authorized action within its bounds (RTA3).
5. Verify – establish what actually happened (RTA4).

Every cycle ends in one of four ways: Continue within the existing mandate, Constrain to narrower bounds, Escalate for additional authority or human judgment, or Terminate.

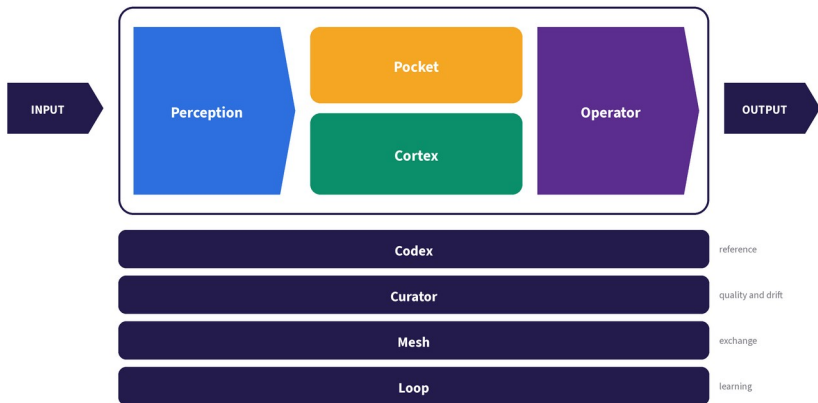
The boundary that matters most lies between Evaluate and Authorize. Interpretation is not authorization: observing and evaluating can become ever more intelligent, and no gain in intelligence creates authority. For example: an agent that concludes that a customer deserves a refund has made a proposal, not a decision. The formula of the architecture fixes this: Intelligence proposes. Authority permits. Mandate bounds. Execution acts. Evidence proves.

The figure shows both loops together. The book sets this out in detail (Chapter 10).



The runtime core: the four RTA layers as a closed control loop, and the Runtime Control Loop that runs across them

2.3 Eight layers of intelligent collaboration



The eight layers of the reference model

The four RTA layers state which questions every action must answer. Eight functional layers, shown in the figure, supply the capabilities that answer them:

- **Perception** – The senses. This layer takes in signals from the environment – requests, events, performance data, documents – and passes them on with their source. Perception is not interpretation.
- **Cortex** – The interpreter. The Cortex translates signals into meaning within the shared vocabulary, composes the context and constructs options. It does not act and it does not authorize.
- **Codex** – The memory. This versioned reference layer holds the shared vocabulary, definitions, decisions, precedents and rules. It is a living institutional memory, not a static dictionary.
- **Curator** – The inspector. The Curator checks quality and consistency across cases, detects drift (behavior or meaning that shifts

unnoticed), flags borderline cases and normative conflicts, and recommends.

- **Pocket** – The capacity manager. The Pocket tracks the available computational, financial and human resources and enforces budgets. A budget here is a constraint with an escalation attached, not a report.
- **Operator** – The executor. Here decisions become actions in the operating environment, and here mandate is enforced most directly: no action proceeds outside the actor's mandate.
- **Mesh** – The connection. The Mesh carries meaning coherently between the Corpora of an organization (section 2.4) and to external parties. For example: a refund issue detected in customer service reaches the Finance Corpus with its context, not only its data.
- **Loop** – The feedback and learning layer. The Loop compares outcomes with intentions and returns the differences as proposed updates. Nothing is rewritten in production simply because the data said so.

The eight layers are not a sequence that every action passes through in a fixed order. They are a thinking model: capabilities that are called upon at different moments, depending on context, risk and mandate, and several can be active at once.

None of the eight layers authorizes. The Curator prepares the decision gate – quality, drift, borderline cases, a recommendation – while the decision itself is taken by the authority function, RTA2, on the basis of mandate and the two normative layers (sections 2.5 and 2.6). Section 4.4 follows one refund request through the whole chain.

2.4 Corpus, MetaCorpus and the container hierarchy



The container hierarchy and the two normative layers

The eight layers say what a governable system must be able to do, not where something happens or under whose responsibility. For that, RTA uses six levels of scope, the container hierarchy: MetaCorpus → Corpus → Domain → Capability → Workflow → Activity. The figure shows these levels, together with the two normative layers of section 2.5.

The MetaCorpus is the organization as a whole. A Corpus is a bounded work area with one coherent mission, such as Service Delivery, People or Market. A Domain is a specialized area within a Corpus; a Capability is a repeatable capacity, such as refund handling; a Workflow is the chain of steps that carries out a Capability; and an Activity is the smallest step within it. Beneath the Activity lies the data it works on.

A Corpus is more than a department. It is a mini-organization with its own mission, scope, vocabulary, norms, work rhythm and learning mechanism, and that gives a synthetic actor the stable context it needs. For example: within one Corpus a term such as “resolved” has one definition, shared by every person and every agent that works there.

There is one Cortex, and it instantiates per scope: the same interpretive function appears at the MetaCorpus, the Corpus, the Domain and the Capability, each time with a narrower scope. Workflows and Activities do not author their own meaning; they inherit it. The organization stays normatively one and operationally differentiated.

The MetaCorpus is also the coordinating layer – as an image, the collective consciousness of the organization. It defines a shared rhythm, coordinates decisions between Corpora that touch one another’s territory, and watches that all Corpora stay aligned with mission, values and legal commitments. It is not a command layer, and it does not replace the board.

Every level has an owner: a person, or a small group of people, accountable for the meaning kept at that level. The Activity is the smallest governable unit, because that is where the question arises: may this actor take this step now? Everything above the Activity is preparation; everything at the Activity has consequences. The book sets this out in detail (Chapter 11).

2.5 CoreSOS and IMUS

A written set of values binds nobody at the moment of action: a synthetic actor that decides at machine speed cannot consult a board. Two

normative layers therefore hold the norms in a form the runtime can consult. They are not among the eight layers; they apply across all of them.

CoreSOS

CoreSOS (Core Semantic Operating System) holds the organization's explicit normative principles, priorities and resolution rules, and makes them available at runtime. It records bounding principles, not detailed steps. For example: "customer interest takes priority over quick processing," or "no AI decision outside an explicit mandate." The Cortex interprets; CoreSOS sets the bounds. Which principles bind the organization is never for AI to decide.

IMUS

IMUS (Integrated Moral-Universal Substrate) is a deliberately small layer of universal, non-negotiable commitments: no serious harm to persons, no deception where people rely on what is said, respect for human dignity. IMUS is not the whole of the system's ethics. It is the commitment that certain things will not happen, even under pressure.

How CoreSOS and IMUS relate

The two work as a pair. CoreSOS handles the normal case; IMUS handles the case in which the organization's own norms have drifted. In a conflict IMUS always takes precedence: CoreSOS may never allow what IMUS prohibits. CoreSOS may be stricter, however: an organization can hold itself to a higher standard than the universal floor.

Neither layer acts by itself. Together with mandate they feed the Action Authorization Decision of section 2.6: CoreSOS and IMUS inform the decision, the authority function RTA2 takes it, and the Operator acts on it.

For example: an agent proposes to settle a complaint by quietly leaving something out – faster for the metrics, against integrity. RTA2 does not allow the action, and the reason is recorded.

CoreSOS is a living layer: it changes as the organization learns. That is also its weakness. Norms can quietly drift away from what the organization would, on reflection, want, and IMUS catches this only where a universal commitment is crossed. The architecture names this as its most serious unsolved governance problem: drift is mitigated, above all by the constitutional moment of section 2.7, not solved.

Norms also come from many levels at once, from society to the individual. Annex 2 describes the ten levels, the rhythmic dimension and the way conflicts between them are resolved. The book sets this out in detail (Chapter 12).

2.6 Mandate: who may do what, and how we know

Access control says who can reach a system, not whether an action is legitimate. That is the task of mandate: the explicit specification of what an actor – human or synthetic – is authorized to do, under what conditions, and where the limits lie. An API token is not a mandate. Every mandate has three parts, the mandate triple:

- **Scope** – What the actor may do: stated positively and finite.
- **Conditions** – Under what circumstances the scope may be used; they must be testable at runtime.
- **Escalation** – What happens at the boundary: to whom the actor hands over, with what context and how quickly.

In RTA mandate is a first-class object: named, stored in the Codex, versioned and enforced at runtime, with a named human owner who can revoke it. If the three parts cannot be stated, the agent should not be deployed.

The Mandate Object is the stored specification. The Mandate Envelope is what the actor may do right now: the bounds resolved at runtime for one actor, one action and one context. For example: a refund agent whose Mandate Object allows 500 euros of refunds a day has already used 420; its Envelope is now at most 80 euros. A better model enlarges what the agent can do, not its Envelope. Capability expansion is not mandate expansion.

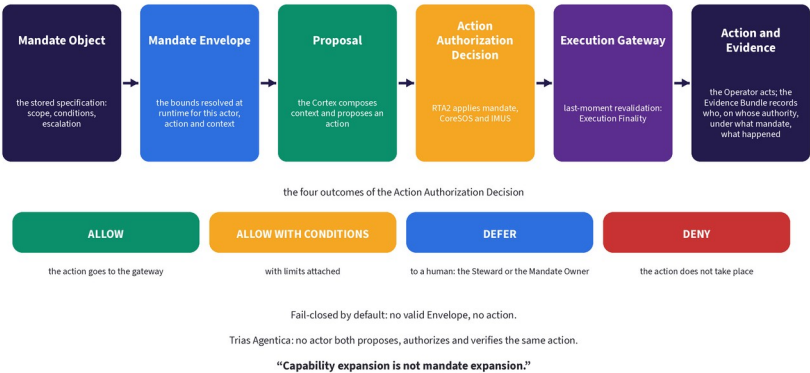
For every proposed action the authority function (RTA2) returns, at the decision gate, an Action Authorization Decision (AAD), based on mandate, CoreSOS and IMUS. It has four possible outcomes:

- **ALLOW** – The action is permitted.
- **ALLOW WITH CONDITIONS** – The action is permitted, provided the stated conditions are met.
- **DEFER** – The decision moves to a human who holds the authority to judge it, such as the Steward (section 3.5). The human in the loop is a structural element, not a fallback.
- **DENY** – The action is blocked.

How much an actor may do without a human is graded on five autonomy levels, assigned per category of action rather than per actor: level 0, observation; level 1, advice; level 2, execution on approval, case by case; level 3, execution within frameworks evaluated in advance; level 4, bounded autonomy within hard limits. The greater the impact of an error, the smaller the autonomous mandate.

Two safeguards complete the mechanics. Execution Finality: authority must still be valid at the moment the action takes effect, because between approval and effect a mandate can be revoked or a limit reached. Trias Agentica: the powers to propose, to authorize and to verify are separated, so that no actor – human or synthetic – proposes, authorizes and verifies the same action.

And how do we know? Every decision records what was decided, on which conditions, why, and under which version of the mandate (section 3.7). The figure follows this path from Mandate Object to evidence. The book sets this out in detail (Chapter 13).



Mandate and the decision gate: from Mandate Object to evidence

2.7 Rhythm: Pulse and Pendula

Rhythm is the most neglected of the three RNS principles (section 1.2). In RTA it is a governance variable: a hard agreement on the calendar, not

goodwill. A rhythm left to goodwill is consumed by daily operations without anyone deciding so. Two concepts carry it:

- **Pulse** – The agreed cadence at which the state of a system is recalibrated against its norms, its goals and its environment: a scheduled moment at which the people concerned decide what has changed and what is updated in the Codex. Without a Pulse, small drifts are absorbed silently; with too frequent a Pulse, the system chases noise. Every Pulse has a named owner, and a missed Pulse is itself recorded.
- **Pendulum** – The deliberate counter-movement against a system's dominant tendency, and above all against over-correction (plural: Pendula). Where speed creates fragility, it is deliberate slowness, like a cooling-off period in law or a buffer stock in a supply chain. Where everyone moves in step, it is deliberate deviation: protected time, free of operations, in which the direction itself can be challenged.

Pendula are hard to keep. Supply creates demand (section 1.3): a Pendulum introduces slack, and slack is exactly what a busy organization consumes first. A Pendulum session is therefore canceled only for safety or continuity, never for busyness. In agentic systems Pendula must be designed in: an agent that is updated continuously on recent feedback is a system without a Pendulum.

The rarest rhythm is the constitutional moment: a scheduled occasion at which the Director role (section 3.5) and the governance bodies ask not whether the actors comply with CoreSOS, but whether CoreSOS still expresses what the organization means.

For example: a weekly Pulse, quarterly Pendula, a continuous Loop and an annual constitutional moment. These cadences are illustrations, not

norms. What the architecture prescribes is that both rhythms exist, each with a named owner, tuned to how fast the environment changes. The book sets this out in detail (Chapter 14).

2.8 Orchestration instead of loose agents

Agents built one at a time, as separate projects, do not scale. Each project clears its own bar and nobody answers for the whole: too many overlapping tools and agents, each with its own assumptions. RTA treats AI as a coordinated ensemble instead. Every module has a clear place in a Corpus, works through the eight layers, and shares the same language, frameworks and goals.

Orchestration does not mean one more software layer on top of the agents. The order comes from the structure they share – mandate, Codex, decision gate and Loop – which bounds every agent, whatever its source.

The agent factory applies production-line thinking to synthetic capacity. What is industrialized is not the agent, which is custom to its workflow, but the process by which agents are designed, built, validated, deployed, monitored and retired. The agent contract makes each agent governable: it states the workflow served, what the agent does and explicitly does not do, its escalation conditions, autonomy level and accountable owner. The control tower watches the whole population of agents – quality, escalations, drift and cost – because agents that each look fine can degrade silently together.

For example: before an agent goes live, a validation gate tests whether what it can technically reach coincides with what it may reach under its mandate.

The technology underneath, the substrate – computing power, data, identity, models and agent runtimes – increasingly arrives as integrated infrastructure from platform vendors, and that is desirable. But a vendor that must remain generic cannot decide what this organization stands for: that is laid down in its own constitution (section 3.2). The substrate may be vendor-provided. The constitution must remain enterprise-owned.

An orchestrated whole is not trusted because it runs smoothly, but because every action leaves evidence. That is the last line of the formula: Evidence proves. The next chapter turns to what an organization must put in place to make this real: the seven deliverables of the Human–Synthetic Enterprise.

3 The Human–Synthetic Enterprise: seven deliverables (What must an organization put in place?)

3.1 From technology to enterprise

Earlier versions of the architecture were designed largely as technical components. Now that platforms supply a growing share of that technology (section 2.8), the center of gravity moves from technology to the enterprise. The question is less how to build the infrastructure, and more how to direct, govern and bound an enterprise that runs on infrastructure delivered by others. Technology determines what is possible. The enterprise determines what of that is desirable and legitimate.

The result is the Human–Synthetic Enterprise: the enterprise in which human and synthetic actors act on behalf of the same enterprise under one governance architecture. It is not a human organization with ever more AI tools added, and it needs no second governance system for AI. Both kinds of actor are identifiable, bounded and traceable; responsibility stays with humans (section 1.3).

This changes the organizing principle. A classical enterprise distributes authority through functions, management layers, procedures and budgets. A synthetic actor has no place in that structure; it has room to act only because the enterprise has granted it. Mandate therefore becomes the primary building block: the enterprise shifts from a function hierarchy to a mandate architecture. Hierarchy does not disappear, but it is no longer the only way authority is distributed.

For example: a new employee gets a job description and a place in the organization chart. A new AI agent gets neither. It gets a named owner, a mandate with limits and a rule for when to hand over to a human.

Seven deliverables follow, each answering one question:

- **I. Enterprise Constitution** – On what ground may anyone, human or synthetic, act on the enterprise’s behalf?
- **II. Mandate and Decision-Rights Architecture** – Who may act, on whose behalf, within which limits and with how much discretion?
- **III. Human–Synthetic Operating Model** – How do constitution and mandate become the actual organization of work?
- **IV. Human–Synthetic Workforce Architecture** – Who should do a piece of work, in which combination of human and synthetic capacity, and who answers for it?
- **V. Semantic and Context Architecture** – Do those who act in the enterprise’s name understand a situation, and the words that describe it, alike?
- **VI. Runtime Governance and Evidence Architecture** – What is an actor permitted to do before it acts, who decided that, and can it be demonstrated?
- **VII. Learning and Constitutional Evolution Architecture** – How does the enterprise change without letting learning silently rewrite what it is?

This is an ordering of the design, not a validated blueprint. But once synthetic actors act at scale, none of the seven can stay implicit: the enterprise could no longer say who acted on its behalf, and why that was legitimate. The figure shows the seven as one whole, from constitution to

evidence, with learning as the way back. The book gives each its own chapter (Chapters 12 to 18).



The seven deliverables of the Human-Synthetic Enterprise

3.2 Deliverable I – The Enterprise Constitution

The Enterprise Constitution is the higher order of the enterprise, from which mandates, policies and instructions derive their legitimacy. The question it answers: on what ground may anyone, human or synthetic, act on the enterprise’s behalf? Among people, much of this could remain implicit. Once synthetic actors act at scale, it cannot.

The constitution is not a workflow or a set of slogans on the wall. It gives direction, not instructions. Its main elements:

- **Purpose and ethical core** – Why and for whom the enterprise exists, and what it holds to be responsible: not only what the law permits,

but what is consistent with human dignity, proportionate, and defensible toward the person affected.

- **Reserved powers and limits of delegation** – Four powers stay with humans, whatever becomes technically possible: changing the fundamental premises; granting, bounding and revoking mandates; resolving normative conflicts that the normative layers cannot settle; and passing final judgment on the legitimacy of what was done. Autonomy may be delegated. Sovereignty may not.
- **Fundamental rights and protected spaces** – The constitution also fixes what the system may not know, infer or institutionalize. The enterprise has a right to the information that execution and governance require, and no automatic right to the inner world of its employees.
- **Deliberative Space** – One protected space: the state before judgment. What people say while they are still thinking does not yet flow into records, profiles, mandates or execution. A person has the right not yet to take a position, and silence has no default meaning.
- **RT-ME (Real-Time Me)** – A second protected space: the personal domain of the individual, which lies outside enterprise authority and which the enterprise recognizes and protects. Personal signals may not be used, unnoticed, for performance management, selection or risk profiling.
- **Reasoning Constitution** – The part of the constitution that states from which foundation the organization reasons when values, interests or goals collide. It does not prescribe what employees must think; it fixes which considerations may not be skipped. Neutral model intelligence does not exist: an organization without a Reasoning Constitution of its own operates under an implicit one,

largely formed by others – the reasoning that comes with the vendor’s model.

The constitution stands at the head of the authority chain: Legitimacy → Authority → Mandate → Authorized Agency → Action → Consequence → Evidence. CoreSOS and IMUS (section 2.5) make it available at the moment of action.

For example: a purchasing agent that keeps running into its order limit may report this and propose a higher limit. It may never raise the limit itself. Granting and widening mandates is reserved to humans.

3.3 Deliverable II – Mandate and Decision-Rights Architecture

The Mandate and Decision-Rights Architecture determines how authority is distributed, bounded, delegated, revoked and evidenced for every actor, human or synthetic. The question it answers: who may act, on whose behalf, within which limits and with how much discretion? Section 2.6 described the mandate itself. This deliverable concerns the organization that mandates create: the mandate network.

The management pyramid solved an information problem: each layer collected, filtered and passed on what the board could not see. A shared Cortex now gives every authorized actor the same context at once. What stays scarce is judgment. Management is not abolished but unbundled: information and analysis move to the Cortex and to synthetic actors; direction, priorities and answerability remain human. The management pyramid organized labor. The mandate network organizes room for action. It has four main elements:

- **An owner at every level** – Delegation runs from the board through every level of the container hierarchy down to the single Activity. Ownership roles such as Mandate Owner, Workflow Owner, Semantic Owner, Codex Owner and Loop Owner are accountability assignments, not positions; one person may hold several. A mandate without an owner is worthless.
- **Escalation as a design feature** – Every mandate has an edge, described as triggers that can be checked at runtime: the action lies outside the mandate, values conflict, context is missing, the impact is high, or a person asks for a human. Escalation is not a failure; it is the design working as intended, and every escalation closes with a named handler and a recorded outcome. Escalations are not incidents; they are missing frameworks.
- **Forbidden without a human** – The list is short and hard. A synthetic actor may not change the frameworks – definitions, policy, thresholds, escalation paths; may not widen any mandate, its own included; may not turn an exception into a rule; and may not decide autonomously about people.
- **Refusal as maturity** – The usual test of an agentic system is what it can do; the more demanding test is what it refuses to do when mandate, context or consequence require refusal. An agent that returns DEFER or DENY with a clear motivation shows the property that matters most. Mature intelligence begins with refusal.

For example: an AI agent may summarize job applications and show how each matches the profile. Deciding who is invited and who is rejected stays with a human who holds that mandate.

The order of work follows: codification comes before automation, because only what is explicitly recorded can be safely delegated.

3.4 Deliverable III – The Human–Synthetic Operating Model

The Human–Synthetic Operating Model is the connecting layer between what the enterprise constitutionally wants to be and how it actually functions. The question it answers: how do constitution and mandate become the actual organization of work? An operating model consists of hard agreements, not intentions. What is not organized does not happen, and what is not protected disappears.

- **Minimum Viable Operating Model** – The smallest set of agreements – on rhythm, roles, decision rights, delegation, work design and measurement – without which the rest of the design does not hold. It is not a reorganization; it is small enough for an existing structure to adopt.
- **Rhythm as a hard agreement** – The rhythms of section 2.7 are put on the calendar and protected: the Pulse of operations, the Pendula of panorama – the long-horizon overview from which direction is set – and the Loop of learning. Measurement follows the same three, because indicators that show only the Pulse pull the whole organization back into operations.
- **Functions and decision rights** – Every core stream names four functions: execution (the Practitioner), semantics and quality (the Steward), panorama and course (the Director) and learning (the Loopkeeper). Left to be done on the side, the last three are not done.

Operations may signal and propose; the Director sets the frameworks.

- **20-40-40** – The design parameter for human work: roughly twenty percent production, forty percent learning and forty percent transformation – turning what was learned into standards and playbooks – so that the organization learns faster than its environment changes. Production is what must still be human: exceptions, relational moments, decisions with moral or strategic weight. The numbers are a design choice, not a measured optimum.
- **Demand engineering** – Supply creates demand (section 1.3). Demand engineering is therefore the deliberate design of what enters a system, at what cadence, under what authorization and with what stop-criteria. It is not the practice of saying no; it is the practice of saying yes deliberately.
- **Agent factory and substrate** – Agents come from the agent factory (section 2.8), not from one-off projects. Before the production line stands the prod-adjacent pilot: a time-bound test with real data and realistic volumes beside production, not in it, judged on criteria defined in advance; missing evidence means No-Go. The substrate – the platform technology underneath – is used wherever it demonstrably meets the requirements; vendor agents work within wrappers and contracts the enterprise controls.

For example: an AI agent takes over a team's weekly reporting. Unless someone decides what the freed hours are for, new requests absorb them. The return on AI is measured not by capacity freed but by capacity protected.

3.5 Deliverable IV – Human–Synthetic Workforce Architecture

The Human–Synthetic Workforce Architecture determines which combination of human and synthetic capacity the enterprise needs, and how agency is divided legitimately between them. The question it answers: who should do a piece of work, in which configuration, and who answers for it? The goal is neither maximal automation nor maximal human control, but the most effective and legitimate division of agency.

The division starts from three categories of work, taken from customer contact, where “agent” means whoever handles the interaction, person or system. The human specialist handles work whose value lies in human judgment, creativity, relational presence or moral attention. With the augmented agent, a human does the work and AI assists by drafting, recalling or flagging; the human owns the final decision. The virtual agent is fully automated, and the Workflow Owner still answers for its outcomes.

For example: a simple account question may move from the human specialist to the augmented agent, and later to the virtual agent. Each step requires explicit promotion: measured quality, an escalation path and an owner who takes responsibility.

Four role profiles carry the human side. They are kinds of work, not job titles, and one person may carry several:

- **Practitioner** – Responsible for execution within frameworks: the operation runs reliably, friction and drift are signaled, and escalation is by rule, not by feeling.
- **Steward** – Responsible for semantic discipline and quality: that the organization means the same thing by its words. The Steward

maintains definitions, classifies new exceptions, may reject AI output on quality grounds and judges the borderline cases that the system defers.

- **Director** – Responsible for panorama, for setting frameworks and for direction within scope. The Director makes trade-offs explicit and protects the Pendula against the operation.
- **Loopkeeper** – Responsible for learning and for securing it: what has been learned becomes standard, findable and trainable.

Each works with a layer – Operator, Curator, Cortex, Loop – and none becomes it: people are not system components. Three further elements complete the deliverable:

- **The buddy agent** – The personal, context-aware synthetic companion of every employee: one recognizable front with many scopes behind it. It presents, routes and explains, and it is never a source of authority: “do this for me” creates no authority that the employee does not hold. Five participation modes – Listener, Adviser, Contributor, Delegate, Escalation – grade its influence; Adviser is the default.
- **The Workbench** – The registered enterprise actor through which a person works with AI: configuration, identity, mandate, memory and connectors, with the buddy agent as its front door. Scope is not access: the rights to read, retain, publish and export are granted separately. One rule covers the whole lifecycle: personal continuity is portable, corporate authority is provisioned, corporate access is revocable.
- **Human-in-the-Learning-Loop** – AI automates first the work through which people used to learn their profession, and an organization that

automates execution faster than it builds new expertise incurs learning debt. This proposed principle therefore asks not only where a human decides, but also where that human develops the competence to bear the responsibility. Automate execution, not the capacity to contradict.

3.6 Deliverable V – Semantic and Context Architecture

The Semantic and Context Architecture determines how human and synthetic actors keep enough shared meaning to act on behalf of the same enterprise. The question it answers: do those who act in the enterprise's name understand a situation, and the words that describe it, alike? Meaning is a governed runtime variable, because an actor can be operationally, legally and technically correct and still be semantically wrong.

For example: a factually correct support answer can land as indifference in a community that is already tense. Beside the question “may this?” a second one appears: “how will this be understood, here?”

Shared meaning fails in recognizable ways. Semantic drift: terms quietly acquire another meaning. Semantic latency: the organization interprets a change too late. Semantic fragmentation: teams use the same words and mean different things. The architecture treats these as signals, not as communication problems. Its main elements:

- **One Cortex, scoped projections** – There is one Cortex (section 2.4); a personal, team or client context is a temporary instance of it. A question starts no unlimited search: it becomes a Context Request; RTA2 answers with a Read Envelope that fixes which sources may be used; and within it the Cortex composes a Cortex projection, the

temporary, scoped and evidenced context for one actor, task and situation. First the permitted reach, then relevance – and access to context creates no right to act.

- **The Codex and canon control** – The Codex contains only what has been legitimated: definitions, policies, thresholds and approved language. Canon control means one active version, no silent updates, and earlier versions kept. Synthetic actors may detect drift and draft updates; they are never the authority over the canon.
- **The Curator function and the Contribution Gate** – The Curator function prepares and recommends; it never decides (section 2.3). What practice yields – a deviating pattern, a proposed definition, a lesson – is candidate knowledge, and it passes the Contribution Gate: humans validate it before it becomes institutional knowledge. Contribution is not canonization.
- **Semantic Integration Module** – Where signals cross a boundary between Corpora, Domains or external parties, this module verifies that the receiver understands a term as the sender does, and that the requested action lies within the receiver's mandate. It never interprets.
- **Nowcasting** – The deepening of Perception that turns incoming signals into short-horizon projections, minutes to a few hours ahead: an estimate of the present, not a prediction of the future.

The aim is not full consensus, but enough shared meaning to stay governable and correctable.

3.7 Deliverable VI – Runtime Governance and Evidence Architecture

The Runtime Governance and Evidence Architecture governs the moment at which an action is about to have a consequence. The question it answers: what is an actor permitted to do before it acts, who decided that, and can it be demonstrated? Its principle is authorization before execution, not explanation afterward.

The decision moment is the point at which interpretation becomes action. Between the two stands an enforced test, the decision gate: may this happen? RTA2 answers with the Action Authorization Decision (section 2.6), and material actions then pass through one enforcement point, the Execution Gateway; section 4.3 follows one case through this chain. Where the evaluation cannot be completed, the default is fail-closed: the action does not proceed. Exceptions must be declared in advance. Routine, low-risk cases take a fast path through pre-authorized patterns.

Logging registers events; evidence makes them accountable. For every material action the enterprise must be able to answer four questions:

- **WHO acted** – Which actor, human or synthetic.
- **ON WHOSE AUTHORITY** – On whose behalf, and through which delegation.
- **UNDER WHAT MANDATE** – Within which mandate, with which context and on which norms, in the versions in force.
- **WHAT ACTUALLY HAPPENED** – Which decision, which consequence, and which human intervention or deviation.

Three objects carry the answers, built during operation and not after an incident: the Decision Record (what was proposed, decided and why), the

Execution Record (what actually happened) and the Evidence Bundle, which connects them into one reconstructable whole.

For example: a regulator asks why a customer's claim was refused. The enterprise does not reconstruct the case from emails; it opens the Evidence Bundle.

Compliance looks back; runtime governance decides forward. Because every step is tested at the moment of action, compliance becomes a property of operation, not a documentation effort afterward. Four articles of the EU AI Act on high-risk systems show the correspondence – a correspondence, not a legal discharge:

- **Article 9, risk management** – A continuous process: risks are identified, weighed against CoreSOS, bounded by mandate and revisited through the Loop.
- **Article 12, record-keeping** – Automatic recording of events: with a reconstructable trace, what an audit asks for is a by-product.
- **Article 13, transparency** – One trace yields the views that a Practitioner, a Steward and a compliance reviewer each need.
- **Article 14, human oversight** – The decision gate, escalation and the kill-switch give people the means to override and interrupt. Overriding must be a normal operation, not a failure to be minimized.

An agentic system is not reliable because it appears to act correctly. It is reliable when it can be shown that it was allowed to act, that it acted within what was allowed, and that what it did was verified against what was authorized.

3.8 Deliverable VII – Learning and Constitutional Evolution Architecture

The Learning and Constitutional Evolution Architecture determines what the enterprise does with the evidence it produces: what may learn by itself, and what may change only through people and procedure. The question it answers: how does the enterprise change without letting learning silently rewrite what it is? Learning must be routed, not merely permitted.

Learning starts in the Loop (section 2.3), from operational evidence: outcomes, errors, overrides and escalations. The Loop learns in two ways. Single-loop learning corrects the action against the existing norm. Double-loop learning asks whether the norm itself has become the problem, and puts that question to the norm's human owners. The two are kept apart; otherwise every override passes as a local correction and the norm quietly follows.

For example: week after week, Stewards override the same rule on compensation. Correcting each case is single-loop learning. Recognizing the pattern, and asking the owner of the rule whether it is still right, is double-loop learning.

What the Loop learns travels one learning chain – observation, classification, assessment, codification, test – and that chain changes what the enterprise knows, not what it permits. What may change, and by which procedure, is fixed in the four tiers of Bounded Constitutional Evolution:

- **Fixed core** – The commitments on which everything rests, such as human dignity and human legitimacy as the source of every synthetic

mandate. It changes only by constitutional ratification, at a constitutional moment (section 2.7).

- **Revisable doctrine** – The strategic and normative interpretations that give direction. It changes through formal governance and is visibly marked as revisable.
- **Operational rules** – Thresholds, workflows, risk classifications and escalation routes. They change by operational adjustment with review, authorized by the competent human owner.
- **Experimental hypotheses** – They may learn automatically, under registration: temporary, bounded, measurable and revocable.

Observation is continuous. Learning is continuous. Constitutional change is discontinuous and authorized. Learning may challenge the governing framework. It may not silently rewrite it.

All of this serves institutional wisdom: the capacity of an organization of human and synthetic actors to act well under uncertainty, at scale and in time, while remaining able to explain, correct and legitimate what it does. Intelligence asks how a goal can be reached; wisdom asks which goal ought to be pursued. For every actor this means connected autonomy: think independently, act constitutionally connected. The architecture cannot guarantee that an enterprise changes wisely. It can guarantee that it does not change silently.

4 RTX – Real-Time Execution (What?)

4.1 From model to practice

Real-Time Execution (RTX) is the application of the architecture in day-to-day operations. Where RTS provides the vision (why we must do things differently) and RTA the structure (how we set it up), RTX shows what happens in practice. It is not an extra layer but an operating paradigm: the execution layer (RTA3) at work. Execution is treated as the point where risk and responsibility become real. Three conditions define it:

- **Gating before execution** – No action takes effect before its Action Authorization Decision has been taken.
- **Bounded units** – Work is done in small steps that can each be tested against mandate.
- **Full traceability** – Every action can be traced to its decision, actor and context.

Real time is a matter of architecture, not of speed. A system is real-time when the decisions about the work – routing, priority, quality, escalation – are taken at the same moment as the work itself. The test is not “how fast does the answer arrive?” but “when did the system decide what to do?” For example: a contact center that samples quality once a week decides a week after the work; one that assesses every reply before it is sent decides in real time.

Authority therefore takes effect at the execution unit, the Activity of the container hierarchy: the smallest action with an effect outside the model. “Issue a refund” is an execution unit; the query that checks the amount is not.

A workflow is the sequence of units that completes a process. Orchestration is not workflow: it composes the units in response to the case – which are needed, under what conditions, and when a flow stops, repeats or escalates. Agents may compose, select and sequence within mandate. They never determine whether an action is permitted.

4.2 Application in various domains

The architecture is deliberately independent of any one domain. Its applications are not separate architectures but configurations of the same one:

- **RTE – Real-Time Enterprise** – The configuration for companies: the Human-Synthetic Enterprise in operation. Its Corpora decide on current signals rather than periodic reports, coordinated by the MetaCorpus. Example: when customer behavior shifts, the Corpus concerned adjusts within its mandate instead of waiting for the monthly report.
- **RTG – Real-Time Governance** – In government, AI already classifies risks in real time while supervision still checks afterward. RTG names AI as an actor with bounded authority and fixes beforehand what it may decide. IMUS weighs heavily here: fundamental rights, due process, democratic legitimacy. Example: when a fraud-detection model flags a citizen, it is settled in advance what the flag may set in motion and when an official takes over.
- **RT-Edu – Real-Time Education** – The emphasis moves from execution to interpretation and guidance, and falls on the Codex and the Curator function. Example: teaching material in a fast-moving

field stays current, while the teacher keeps room to depart from the standard pattern when a student needs it.

- **RT-Health – Real-Time Health** – Patient safety, informed consent and privacy put heavy demands on every layer. CoreSOS holds the clinical protocols, IMUS the universal medical-ethical commitments. Example: patient monitoring is linked to AI triage, and when a safety threshold is approached the Pocket enforces escalation to a clinician without delay.
- **RT-NGO** – Civil-society organizations decide under uncertainty: context is incomplete, and local and international norms can conflict. The configuration is therefore more conservative: less automation, more escalation and a larger role for Stewards. Example: in emergency aid the facts are established first; allocation follows.
- **The private sphere: RT-Family and RT-ME** – In the household, AI may support but never replace human responsibility, moral judgment or closeness; a parent, not a system, keeps watch. RT-ME, the personal domain of the individual, lies outside enterprise authority: this is where the architecture's authority ends. Example: an employee's remark about being tired stays personal; the organization does not use it.

The pattern does not vary: the four RTA layers, the eight layers, the container hierarchy, Pulse and Pendula, layered normativity. The calibration does: which layers carry most weight, how cadences are tuned, how heavily IMUS is loaded, what CoreSOS contains and what a mandate covers.

The configurations are not equally ready: beyond the enterprise they are conceptual extensions. In high-risk domains the consequences are

heavier, the regulation is stricter and professionals answer to their own professional bodies. The book sets this out in detail (Chapter 19).

4.3 Practical example: a refund request through the chain

A good first decision to bring under the architecture is the refund: a clear decision moment with a direct financial effect. The thresholds in this constructed example are illustrative: a refund up to 100 euros is automatic, one up to 300 euros is allowed with conditions, and a larger one is escalated. This is how one request moves through the chain:

1. A customer in the United States writes in Spanish to an operation run from Europe and asks for a refund of 250 euros. The wording signals frustration and legal sensitivity. The Perception layer registers the ticket and nothing more.
2. The Cortex classifies the request and proposes a scenario; a translation component renders it in the operation's working language.
3. Each activity – classifying, translating, retrieving, drafting, authorizing – receives only the context it needs: its own Read Envelope and Cortex projection. A retrieval component fetches the refund policy with its source status.
4. A drafting component writes a reply that includes the refund. This is a proposed action, nothing more. The Curator layer assesses the draft, but a high quality score is not a permission.
5. At the decision gate, mandate, CoreSOS and IMUS are brought together in one Action Authorization Decision. On amount alone, 250 euros would be ALLOW WITH CONDITIONS. Because of the risk signals the outcome is DEFER: the synthetic actors may prepare, not execute.

6. The Steward, the human with authority at this level, receives a targeted handover and approves. The approval is bound to this proposal, this projection and this version of the mandate.
7. The approved action passes through the Execution Gateway, the single enforcement point. It revalidates account and amount at the last moment (Execution Finality) and grants the refund tool a minimal entitlement, for this transaction only. The reply is rendered in the customer's language.
8. The Decision Record and the Execution Record enter the Evidence Bundle. Verify finds the refund equal to what was authorized and sets Continue. The Loop receives the outcome and reads the patterns across many such cases; what then changes is for the owners to decide.

In short, the components prepare, the gate decides, a human judges what is deferred, and the evidence shows who acted, on whose authority, under what mandate and what actually happened. The chain does not guarantee good decisions. It guarantees that a decision can be located, bounded and reconstructed.

4.4 From pilot to operating model

Many organizations understand that AI needs governance and still fail, because they try to solve everything at once. The right approach is the opposite: not broad but narrow, not perfect but concrete. A mandate architecture does not begin with a platform. It begins with one decision. The roadmap has six steps:

1. Choose one real decision moment – a refund, a moderation decision, a customer answer – where it is clear what happens, who is involved and what the risk is.
2. Make the mandate for that decision explicit, and deliberately simple: what is permitted, what is not, and what happens in case of doubt.
3. Enforce it: the action passes the decision gate before it proceeds. The system changes from an advice machine into a controlled system.
4. Organize escalation as part of the process, not as an exception. Many early decisions escalate; without enough Steward capacity the system breaks.
5. Record everything: not only the outcome of each decision but why it was taken, in which context and under which rule.
6. Learn and adapt: where the system does not fit, humans adjust the mandate. The system grows through better bounding, not through new technology.

Only when one use case works does extension begin, by reuse rather than rebuilding. Pilots run beside production (section 3.4) and move on only after a Go decision based on a full set of evidence. A successful demonstration is not enough.

Measurement comes first. Before anything is scaled, a baseline is recorded and a measurement framework is built, so that change can be attributed to the architecture and not to its surroundings. This evidence program measures, among other things:

- **Traceability** – Can decisions be reconstructed?
- **Reproducibility** – Are outcomes consistent under similar conditions?

- **Escalation correctness** – Does escalation happen where it is required?
- **Decision Lead Time** – How long does it take from the first detectable signal of a development to the decision of the authorized actor?

The greatest risk is haste. The first case works, and the pattern is applied everywhere at once, which produces complexity, inconsistency and loss of control. A mandate system grows through repetition, not speed: autonomy is raised step by step, per type of action and risk class, never across the board. AI may never grow faster than its governance. The book sets this out in detail (Chapter 21).

4.5 Conclusion: RTS, RTA and RTX together

The refund example illustrates how vision, architecture and execution come together. RTS provides the philosophy and direction – the why, built on rhythm, norms and semantics. RTA delivers the runtime backbone – how state, authority, execution and accountability are organized. RTX shows the implementation in practice – what happens in a given context. Together they form RTSAX, the architecture of the Human–Synthetic Enterprise. Machines may make the first move. Humans remain in the driver’s seat: they give and safeguard meaning, and they grant, bound and revoke the mandate.

Four propositions sum up what this booklet has tried to say:

- **Rhythm is a governance variable** – Most frameworks treat governance as a static set of constraints; RTSAX treats it as a living cadence.
- **Where normative reasoning lives is a positional choice** – For platform vendors it is right to keep it outside the system; for

organizations that operate close to their own decisions it is right to bring it inside.

- **Measurement comes before architecture** – Build the measurement infrastructure before choosing the architecture, because the value of an architecture cannot be assessed without it. Its absence is, on our reading, one of the principal reasons most agentic-AI investments produce no measurable return.
- **Mandate comes before capability** – No actor may act beyond its mandate. Capability expansion is not mandate expansion.

The honest position is that RTSAX is a design hypothesis, not yet validated. It is being applied, and the collection of evidence is at an early stage: a defensible claim about its value requires at least twelve to eighteen months of measurement after deployment, against a controlled baseline. It is therefore offered as a working hypothesis rather than a finished prescription – open to correction, to disagreement and to the empirical test it should be held to.

Annex 1 – Glossary of terms

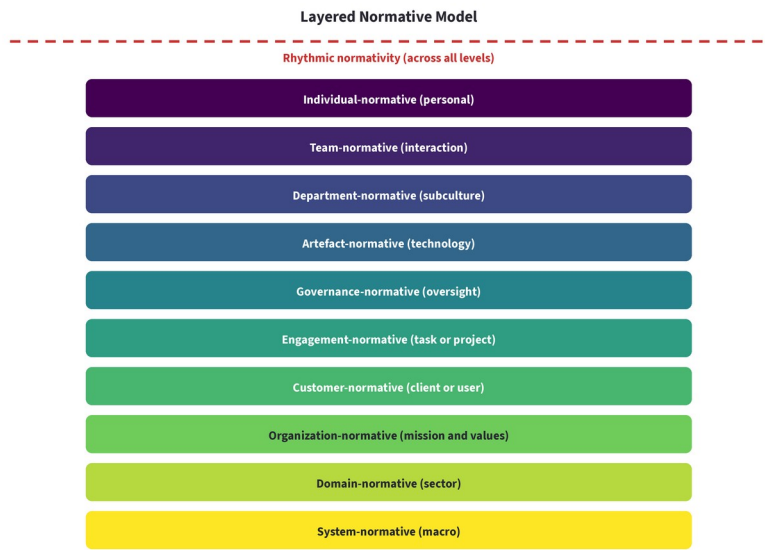
- **RTSAX** – Real-Time Society, Architecture and Execution: the name of the whole, the architecture of the Human–Synthetic Enterprise.
- **RTS** – Real-Time Society, the Why: rhythm, norms and semantics as the load-bearing functions of any social order under stress.
- **RTA** – Real-Time Architecture, the How: the runtime core of four layers that work as a closed control loop.
- **RTX** – Real-Time Execution, the What: a way of operating in which every action is authorized first, bounded and traceable.
- **RTA1–RTA4** – The four RTA layers: State, Authority, Execution and Accountability.
- **Runtime Control Loop** – Observe → Evaluate → Authorize → Act → Verify, ending in Continue, Constrain, Escalate or Terminate.
- **RNS triad** – Rhythm, Norms, Semantics: the three functions every stable system must serve. Ethics lives within norms.
- **Human–Synthetic Enterprise** – The enterprise in which human and synthetic actors act for the same enterprise under one governance architecture.
- **Seven deliverables** – What a Human–Synthetic Enterprise must put in place: constitution, mandate, operating model, workforce, semantics, runtime governance and evidence, learning.
- **Eight layers** – Capabilities, not a sequence: Perception (senses), Cortex (interprets), Codex (remembers), Curator (inspects and recommends), Pocket (allocates resources), Operator (acts), Mesh (connects), Loop (learns).

- **Corpus and MetaCorpus** – A Corpus is a bounded work area with one coherent mission. The MetaCorpus is the enterprise level that keeps the Corpora coherent.
- **CoreSOS** – Core Semantic Operating System: the organization's explicit normative principles, priorities and resolution rules, available at runtime.
- **IMUS** – Integrated Moral-Universal Substrate: the deliberately small layer of universal, non-negotiable commitments that takes precedence over CoreSOS.
- **Layered normativity** – Ten levels of norms, from universal to individual, crossed by the dimension of rhythm (Annex 2).
- **Mandate** – What an actor, human or synthetic, is authorized to do, under what conditions and with what limits: scope, conditions, escalation.
- **Mandate Object and Mandate Envelope** – The stored specification of a mandate, and the bounds resolved from it at runtime for one actor, action and context.
- **Action Authorization Decision (AAD)** – What the authority function (RTA2) returns on every proposed action: ALLOW, ALLOW WITH CONDITIONS, DEFER or DENY.
- **Autonomy levels** – Five levels, 0 to 4: observation, advice, execution on approval, execution within frameworks, bounded autonomy.
- **Execution Finality** – Authority must still be valid at the moment the action takes effect.
- **Trias Agentica** – No actor both proposes, authorizes and verifies the same action.

- **Pulse and Pendula** – The agreed cadence of recalibration, and the deliberate counter-movements against over-correction and a dominant tendency: deliberate slowness or deliberate deviation.
- **Constitutional moment** – The rare, scheduled Pulse at which the norms themselves are examined.
- **Enterprise Constitution** – The higher order of the enterprise: purpose, ethical core, reserved powers and the limits of delegation.
- **Reasoning Constitution** – The part of the Enterprise Constitution that fixes from which foundation the organization reasons.
- **Deliberative Space** – The protected state before judgment: contributions are provisional, and silence has no default meaning.
- **RT-ME** – Real-Time Me: the personal domain outside enterprise authority, which the enterprise recognizes and protects.
- **Practitioner, Steward, Director, Loopkeeper** – The four human role profiles: execution within frames; semantic discipline and quality; direction; securing what is learned.
- **buddy agent** – Every employee's personal synthetic companion and interface to the architecture; not a source of authority.
- **Workbench** – The registered enterprise actor through which a person works with AI.
- **Cortex projection** – The temporary, scoped context the Cortex composes for one actor, task and situation.
- **Contribution Gate** – The gate that candidate knowledge passes before it becomes institutional knowledge. Contribution is not canonization.

- **Semantic Integration Module** – The verifier at boundaries: it checks that both sides understand a term alike. It never interprets.
- **Evidence Bundle** – The record that joins decision and execution into one reconstructable whole.
- **Bounded Constitutional Evolution** – The four-tier change regime: fixed core, revisable doctrine, operational rules, experimental hypotheses.
- **Institutional wisdom** – An organization's capacity to act well under uncertainty while able to explain, correct and legitimate what it does.
- **Semantic sovereignty** – Human control over meaning-making in interaction with AI. For the individual: consent to assistance is no consent to profiling.
- **Real-Time Intelligence** – The capacity of the whole system, not of any single layer, to perceive, interpret, decide within mandate, execute and learn.

Annex 2 – Layered normativity: from universal to individual



The layered normative model: ten levels and the rhythmic dimension

Normativity (norms and values) is not one thing. Where traditional organizations treat ethics as something external, compliance rules placed on top of processes, RTSAX builds the norms into the design. The figure shows ten levels, roughly from universal to individual, plus one dimension that crosses them all: rhythm. The model is a discipline for seeing how norms combine and where they collide, not a claim that every organization has exactly ten levels.

The ten levels, from macro to micro

- **System-normative (macro)** – The highest framework: fundamental rights, legislation and broadly shared social norms. IMUS carries its non-negotiable core, so that it is present at the moment of action. Example: a reply that would deceive a customer in a safety-critical situation is denied, whatever the targets say.
- **Domain-normative (sector)** – The standards and good practice of a sector: privacy and safety in healthcare, anti-fraud rules in finance, pedagogical guidelines in education. Example: in RT-Health the threshold for AI autonomy is higher; in case of doubt a human is always consulted.
- **Organization-normative (mission and values)** – The mission, values and strategic choices of one organization. CoreSOS makes them available at runtime as principles, priorities and resolution rules. Example: if the value is “customer first, even at the expense of short-term profit,” customer interest weighs heavier wherever it conflicts with margin.
- **Customer-normative (client or user)** – The norms and expectations of those the organization serves: service agreements, contractual requirements, cultural sensitivities. The Codex records them within the broader frames. Example: “for customer X, quotes above amount Y always get a human double-check”; such a quote is deferred to a person.
- **Engagement-normative (task or project)** – Temporary norms for one project, assignment or crisis. They are loaded for the duration and unload when the engagement ends; the Loop then evaluates

whether they were met. Example: a stricter confidentiality regime for one client project.

- **Governance-normative (oversight)** – The frameworks of corporate governance, internal audit and external regulators. They hold as runtime constraints, not as findings in a later audit. Example: “transactions above €10,000 must be approved by a human” becomes a condition in the mandate; such a transaction is deferred to a person.
- **Artefact-normative (technology)** – Every technology choice carries values: “code is law.” The choice is therefore made deliberately: privacy by design, bias detection, explainability. Example: a draft reply with offensive language is flagged at the Curator layer and denied at the decision gate before it reaches the customer.
- **Department-normative (subculture)** – Each Corpus develops its own way of working. That is allowed, but the Mesh keeps a Corpus from becoming an island. Example: each Corpus periodically reviews whether its decisions still fit the company values.
- **Team-normative (interaction)** – The daily, informal norms of small groups: how colleagues decide, meet and give feedback. Through the Loop a team sees the effects of its decisions and may adjust its rhythm within the frames. Example: “no meetings after 3 pm,” because the data showed late meetings to be less productive.
- **Individual-normative (personal)** – Every person has their own beliefs, values and limits. Important decisions are therefore explainable, the actions of AI transparent, and objection and human intervention always possible. Example: an employee who finds a decision wrong can raise the alarm through an escalation.

- **Rhythmic normativity (across all levels)** – Not an eleventh level but a dimension: the weight of a norm can shift with time. A norm may be foregrounded for a declared period and then released: a Pendulum at the normative level. Example: during an innovation sprint, efficiency gives way to creativity. Critical decisions can carry a mandatory pause, such as a 24-hour reflection period, and a periodic ethical audit asks whether the norms need recalibration.

When levels conflict

The levels do not always agree. A team norm (“we respond quickly”) can clash with a customer norm (“this customer needs a careful explanation”). People settle such conflicts by judgment; a synthetic actor lacks that implicit weighing, so the architecture makes it explicit in three ways:

- **Precedence** – Higher levels generally override lower ones, and IMUS takes precedence over CoreSOS.
- **Explicit rules** – Where levels of comparable height conflict, CoreSOS records the rule, for example “patient safety takes precedence over throughput.”
- **Escalation** – What neither resolves goes to the responsible human owner, whose decision is recorded for similar cases.

In this way conflicts are resolved by transparent rules or escalations, not by silent compromise. Within what access rights permit, the Weighted Semantic Authority Model gives each contribution, human or synthetic, an explicit weight; it is an influence system, not a truth system. Human responsibility is not taken away; it is made explicit and scalable. The book sets this out in detail (Chapter 12).

Annex 3 – What has changed since version 1.0

- **The name** – Version 1.0 called the whole “RTS.” The whole is now RTSAX; RTS is the Why, RTA the How and RTX the What.
- **RES becomes RNS** – The RES triangle (Rhythm, Ethics, Semantics) is now the RNS triad: Rhythm, Norms, Semantics. Ethics lives within norms.
- **RTA² is retired** – Runtime assurance is now a pattern across the four RTA layers and part of deliverable VI. The layers are written RTA1 to RTA4, without superscripts.
- **Corpus and MetaCorpus** – Both now sit in a six-level container hierarchy: the MetaCorpus at the top as the enterprise level, the Corpus as a mission-level unit within it.
- **The eight layers** – The same eight, always listed in the same order. One correction: version 1.0 let CoreSOS block actions and the Curator check and adjust them. The Curator layer now prepares and recommends, the authority function (RTA2) decides and the Operator acts.
- **CoreSOS and IMUS** – Their relation is unchanged: IMUS takes precedence over CoreSOS. IMUS is now deliberately small, and both feed the Action Authorization Decision.
- **New** – The four RTA layers and the Runtime Control Loop; mandate as a first-class object, with the Action Authorization Decision; the Human–Synthetic Enterprise and its seven deliverables; the human roles and the buddy agent; evidence and compliance as runtime properties; Bounded Constitutional Evolution.

- **Status** – The vocabulary of this edition is a proposed canon, pending the author’s ratification.
- **Open decisions** – Still to be ratified, among others: the relation between CoreSOS and IMUS; the powers of the Curator function; whether a buddy agent may hold a mandate of its own; the autonomy and access scales; and the RT-ME boundary.
- **The full argument** – The book (second edition) gives the full argument, and Annex 4 now follows its References; the consolidated working paper (v41) holds the full text and the register of open items.

Annex 4 – Literature

Chapter 1: RTS – Real-Time Society (Why?)

Core topics: acceleration and meaning, AI as synthetic actor, semantic direction, the governable enterprise.

1. Rosa, Hartmut (2013). *Social Acceleration: A New Theory of Modernity*. New York: Columbia University Press.
2. Latour, Bruno (2005). *Reassembling the Social: An Introduction to Actor-Network-Theory*. Oxford: Oxford University Press.
3. Luhmann, Niklas (1979). *Trust and Power*. Chichester: John Wiley & Sons.
4. Scott, James C. (1998). *Seeing Like a State: How Certain Schemes to Improve the Human Condition Have Failed*. New Haven: Yale University Press.
5. Rodrik, Dani (2011). *The Globalization Paradox: Democracy and the Future of the World Economy*. New York: W. W. Norton.
6. Parkinson, C. Northcote (1955). *Parkinson's Law*. *The Economist*, 19 November 1955.
7. Mayer, R. C., Davis, J. H., and Schoorman, F. D. (1995). An integrative model of organizational trust. *Academy of Management Review*, 20(3), 709–734.
8. O'Neill, Onora (2002). *A Question of Trust (Reith Lectures)*. Cambridge: Cambridge University Press.

Chapter 2: RTA – Real-Time Architecture (How?)

Core topics: control loops, rhythm, accountability, measurement.

1. Ashby, W. Ross (1956). *An Introduction to Cybernetics*. London: Chapman & Hall.
2. Beer, Stafford (1972, second edition 1981). *Brain of the Firm*. Chichester: John Wiley & Sons.
3. Boyd, John. The OODA loop, as reconstructed in Osinga, Frans P. B. (2007). *Science, Strategy and War: The Strategic Theory of John Boyd*. London: Routledge.
4. Meadows, Donella H. (2008). *Thinking in Systems: A Primer*. White River Junction, VT: Chelsea Green.
5. Bovens, Mark (2007). Analysing and assessing accountability: a conceptual framework. *European Law Journal*, 13(4), 447–468.
6. Roberts, John (1991). The possibilities of accountability. *Accounting, Organizations and Society*, 16(4), 355–368.
7. Goodhart, Charles A. E. (1975). Problems of Monetary Management: The U.K. Experience. In *Papers in Monetary Economics*, Reserve Bank of Australia.

Chapter 3: The Human–Synthetic Enterprise: seven deliverables (What must an organization put in place?)

Core topics: learning, safety, agentic-AI security, compliance as a runtime property.

1. Argyris, Chris, and Schön, Donald A. (1996). *Organizational Learning II: Theory, Method, and Practice*. Reading, MA: Addison-Wesley.

2. Hollnagel, Erik (2014). *Safety-I and Safety-II: The Past and Future of Safety Management*. Farnham: Ashgate.
3. Willison, Simon (2024–2026). Selected blog posts and conference talks on agentic-AI security, including the formulation of the “lethal trifecta”. simonwillison.net.
4. European Union (2024). Regulation (EU) 2024/1689 (“Artificial Intelligence Act”). Official Journal of the European Union, L series, 12 July 2024.
5. European Union (2016). Regulation (EU) 2016/679 (“General Data Protection Regulation”). Official Journal of the European Union, L 119/1, 4 May 2016.

Chapter 4: RTX – Real-Time Execution (What?)

Core topics: domain configurations, capacity and demand, interoperability standards, the implementation roadmap.

1. Bos, David (2024). *Transforming Customer Service: How GenAI Reduces the Need for Native Language Agents*. 5CA, August 2024.
2. Bos, David (2025a). *Supply Creates Demand: A Perspective on Capacity in Modern Organizations*. 5CA, January 2025.
3. Bos, David (2025b). *From Human Doing to Human Being – with AI as an Enabler*. 5CA, December 2025.
4. Bos, David (2026a). *Designing for AI Is Half the Job*. 5CA, May 2026.
5. Bos, David (2026b). *The Market Priced the Seat. It Has Not Priced the Judgment*. 5CA, July 2026.
6. Anthropic (2024). *Model Context Protocol (MCP)*. Open-source protocol specification, available at modelcontextprotocol.io.

7. Agent-to-Agent (A2A) Protocol. Originally developed by Google and announced in April 2025; subsequently transferred to the Linux Foundation.
8. Microsoft (2025). The vendor's organizational framing of the Frontier Firm; with the platform documentation published as Microsoft Learn (2026).

Source documents

1. van Haaren, Otto Th.G. (2025). RTS – Real-Time Society: Architecture for a Meaning-Driven AI Society. Booklet, version 1.0, September 2025.
2. van Haaren, Otto Th.G. (2026). RTSAX – The Architecture of the Human-Synthetic Enterprise. Second edition, October 2026 (first edition: Real-Time Architecture, September 2025), co-written and edited with Claude (Anthropic) and ChatGPT (OpenAI).
3. van Haaren, Otto Th.G. (2026). RTSAX – The Architecture of the Human-Synthetic Enterprise. Consolidated working paper v41, September 2026, co-written and edited with Claude (Anthropic).

Intelligence provides capability. It does not confer a mandate.

When AI acts on an enterprise's behalf, authority, boundaries and accountability must be explicit. Governance must operate during execution, while people retain the understanding and means to direct and intervene.

This is the premise of the Human-Synthetic Enterprise: people and synthetic actors working together within a shared governance framework.

What kind of enterprise do we want to be when synthetic actors can also act on our behalf?

Otto Th.G. van Haaren

