AN INDEPENDENT THESIS

A brain for *every business.*

Coordinate Operations Architecture

ABDUL NIHAF

People should not have to read your mind.
The business can remember, guide the work and check the result.

2 October 2026 Revised 3 October · Version 1.3 About 50 minutes

THE WHOLE IDEA, IN ONE LOOP

From what you want to what actually happens.

Make your expectation clear in the work. Give people a step they can see. Let the result tell you what needs fixing.

01 · DIRECTION

You choose the goal.

What should happen? What matters?

02 · CONTEXT + AI

The system works out the next step.

It uses the facts, rules, permissions and history.

Keep the context current.

03 · PERMISSION

The rules check what may go ahead.

Block the action if it is not allowed.

04 · ACTION

A person or tool does the work.

Give the right person, software or device a clear task.

05 · PROOF

Check what actually happened.

Did the work achieve the intended result?

Bring the result back. Update the facts. Use them to choose the next allowed step.

◇ **Something missing, wrong or not approved?** Pause that action. Check the facts or reach the person who can decide.

The same loop can guide physical work, digital services and software. Each business needs its own knowledge, permissions and checks. The “brain” is the whole connected system, so no person has to guess the parts that were left unsaid.

An independent conceptual thesis · Not peer reviewed

THE IDEA IN A MINUTE

How much disappointment begins with “they should have known”?

You ask someone to take care of something. In your mind, you can see what “done properly” means: the promise to keep, the small detail to notice, the mistake to avoid. They cannot see that picture. They do the work as they understand it. When the result comes back wrong, you feel let down. They feel blamed for something they did not know.

Now there are two things to repair: the work and what happened between you. You might carry that feeling home. They might go home thinking that whatever they do will never be enough. A business can make ordinary people feel this way every day without anyone intending it.

What if the business could hold more of that picture, so neither person had to guess?

The idea is simple. Build a system that knows the work, the rules, the people and what has happened so far. Let AI use that knowledge to work out the next step. Check that the step is allowed. Give a person, a piece of software or a device a clear task. Then look at the result before saying the work is done.

This is the idea behind **Coordinate Operations Architecture, or COA**. More of the work of remembering, connecting and checking can move from someone’s head into the way the business runs. You can still care deeply about the result. You no longer have to hope that another person carries your whole picture of it in their head.

The ambition reaches across any kind of business: a shop, a workshop, a hotel, a clinic, a software company or a digital service. It also reaches through each business, from setting its direction to buying, selling, paying and doing the daily work. Each needs its own knowledge. The way that knowledge connects to action can follow the same pattern.

That whole system is what this thesis calls a **business brain**. AI is part of it. So are the records, the rules, the tools and the checks. No single model has to know everything. What matters is that the right knowledge can reach the right task when it is needed.

This is a proposal for how to build and test that arrangement. It does not claim that every use described here has already been built or proved. The examples show how the idea works; they are not results from an experiment.

THE CENTRAL IDEA

Put the expectation where the work happens.
Make the next step clear.
Check the result before blame begins.

---

01 · THE PROBLEM

Why does so much work come back to the person who knows?

Someone is ready to send a piece of work. But is it the right version? Is this the person it was meant to go to? Who said yes? A workshop, a clinic and a software team may handle different things, but each can reach the same pause.

So they ask the person who knows. That person gives an answer, and the work moves again. A few minutes later, another question arrives.

Can this customer get a discount? Has the payment come through? Is the room ready? Which promise comes first? Each question is small. Together, they show how much of the business lives inside someone's head.

The records hold pieces of the answer. One message has the promise, another has the change, and someone remembers why the usual rule does not apply. The owner, manager or experienced worker joins those pieces for everyone else. The business appears to run on its tools, but that person supplies the missing connections.

Better training can help. So can better people and clearer messages. Yet the next person still has to know what to ask, where to look and when an answer might be wrong. You cannot ask about a detail you do not know matters.

Take a familiar instruction: "Finish this and send it." To the person giving it, the meaning may seem obvious. To the person receiving it, there are gaps. Finish which version? Against which promise? Send it to whom? Does it need approval? How will anyone know that it arrived?

The instruction feels simple because someone else is expected to carry what it leaves out. In a physical business, that gap may become a question across a room. In a digital business, it becomes another message or another task waiting for a reply.

After enough of these moments, people begin to expect disappointment too. The owner checks every detail because trusting the process feels risky. The worker asks before making even a small choice because being wrong feels worse than waiting. The business becomes slower, but the deeper cost is that people start guarding themselves around one another.

COA starts by bringing those missing pieces into the system. What does it need to know before the work can move? Which choices must it block? What should happen if the facts do not agree? When those answers are built into the work, fewer people have to keep rebuilding them from memory.

02 · THE CHANGE

If someone has to read your mind, disappointment is built in.

There is nothing wrong with expecting good work. A customer should expect a promise to be kept. A worker should expect a safe task and fair treatment. The trouble begins when the expectation lives only in one person's head and another person is judged against it.

When you care about the work, part of you wants to trust the person doing it. Another part keeps watch because you know what could go wrong. After enough missed expectations, the watching can begin to feel safer than the trusting. The aim here is to move some of that watchfulness into the design of the work, where it can be shared and improved.

That is the human meaning of **zero hidden expectation**. Put what matters into the work itself. Do not expect someone to use a fact the task never gave them, remember an exception from last month, or notice a risk the system kept out of sight. Do not expect an AI agent to know a new rule because another agent once read it.

"Humans are dumb executors" was the blunt starting phrase behind this thesis. Taken as a judgement about people, it misses the point. People bring skill, care and the ability to notice what a system has missed. The change is to stop making them carry hidden business logic as a condition of doing a simple step well.

Imagine two items with almost the same name. A worker picks the wrong one. It is easy to say they should have paid more attention. But why did the task offer the wrong item as a valid choice? Why was there no check before the choice changed the stock or reached the customer?

A person's conduct can still matter. The first explanation should not be their carelessness when the system left an avoidable trap in their path.

The owner can ask a more useful question: *What did I expect that the work never made clear?* If the wrong choice was possible, fix the choice. If a fact was missing, connect it. If the result was never checked, add the check. A failure becomes something to examine and improve, instead of another reason to feel disappointed in a person.

This does not remove the need for skill. Someone using a machine still needs to know how to use it safely. A clinician still needs the right training and authority. The system can organise the work around that expertise. A confident instruction cannot supply expertise that is missing.

What it can supply is a clear next step: here is the thing, here is what to do, here is what a good result looks like, and here is how to stop if something is wrong. The person can focus on that work without carrying every rule and every promise that led to it.

The visible task can be simple because the system behind it holds the detail. Simplicity has to be built somewhere.

This does not mean no one will ever let you down. People, AI, suppliers and the world can still fail in ways you did not cause. It means the business takes responsibility for the expectations it can make clear, the mistakes it can prevent and the failures it can see. That is a more honest kind of control than hoping everyone will always know what you meant.

> Design for the honest mistake made by someone who does not yet know what they are missing.

THE HUMAN DIFFERENCE

The same work, with the expectation in a different place.

When it stays in a head	When the business carries it
01 "They should know."	01 Say what matters and why.
02 They have to guess.	02 Give a clear, allowed step.
03 The result misses the mark.	03 Check the real result.
04 Both people feel let down.	04 Fix the process when it fails.

This is a design argument, not a claim that disappointment can be eliminated. It shows how an avoidable source of blame can become a visible problem to solve.

03 · CONTEXT

Knowing the business means knowing what matters now.

You can tell AI a great deal about a business and still get the wrong next step. The description may be correct in general. But the order changed this morning, the approval ran out yesterday, or the room still has a fault that nobody has recorded.

That is why context means more than a long prompt. It means the facts around this task, connected to the place they came from and kept up to date.

Start with the thing itself. Which order, person, room, file or software version are we talking about? What does it belong to? An order belongs to a customer and a promise. A file belongs to a piece of work and a version of the brief. In the architecture, these identified things are called **entities**.

Next, where does the thing stand? An order waiting for approval is different from one ready to send. A room being cleaned is different from one checked and ready for a guest. This is its **state**. The state changes what can happen next.

Then come the rules. Who can approve the next step? How much may be spent? Which checks must pass? What was agreed before, and has anything changed since? The system needs the relevant history as well as the current answer.

It also needs to know what it does not know. “Ready yesterday” does not mean “ready now.” A person’s report, a sensor reading and a confirmed payment each support different conclusions. A useful record says where the fact came from and when it was checked.

Compare “send the finished work to the client” with “send the approved third version to the named client, once these checks pass.” The second instruction is useful because the pieces are connected. It still needs a final check that the brief, version and approval remain current.

That is the difference between “I told them” and something the next person can actually see when the work reaches them.

None of this means giving every AI call every file the business owns. A purchasing task needs the facts that affect that purchase. It does not need private staff records. The aim is enough reliable context for this decision, with a way back to the source when more detail is needed.

FIGURE 1 · WHAT THE SYSTEM NEEDS TO KNOW

Six questions make the next step clear.

What is it?

The exact thing and what it belongs to

Where does it stand?

Its stage now, and when that was checked

What is allowed?

Rules, limits and who can approve it

What should happen?

The goal and the next step

What already happened?

Earlier actions and why they happened

What is still unknown?

Missing facts and cases that need help

↓

Enough context to choose the next step

Know where the facts came from, how recent they are and what is still missing.

↓

AI can use these facts to choose or suggest a next step.

A clear instruction rests on connected, current facts. A longer prompt alone cannot keep those facts up to date or enforce permission.

04 · COORDINATES

Before deciding what to do, be clear about what you are doing it to.

“Send the final version” sounds precise until two files are both called final. “Get the room ready” leaves room for error if the room number or booking is unclear. A useful instruction starts by removing that doubt.

A coordinate is an address for the work. It tells the system exactly which business, task, thing and situation it is dealing with. The word comes from the same everyday need as an address on a parcel: a correct action sent to the wrong place is still wrong.

The address changes with the business. A hotel needs the property, room and stay. A clinic needs the correct person, journey and station, with the right controls over that information. A software release needs the code version and the place where it will run. Each action must point to the exact thing it will affect.

This gives a useful order for building: **entity, then action, then event**. First, identify the thing. Next, decide what may be done to it and what must be true before that happens. Then keep a record of what actually happened. A button should come after its meaning is clear.

That record is the **event**. It should say more than "done." What changed? Who or what changed it? When? What permission applied? What checks passed, and what result was seen? If the record later needs correcting, keep the earlier entry and the reason for the correction.

COA calls a defined part of this work a "closed" space when the allowed choices are known and the system blocks the others. For example, a task for one approved version should not be able to send a different version by mistake. The same rule must hold wherever that action can be taken.

The world itself stays open. A new request, a damaged item or two conflicting records can fall outside what the system knows. That case needs a place to go: pause, check, clarify or ask the right person. It must not be forced into the nearest available choice.

There is still room for ordinary language. A worker may need free text to describe a strange noise or an unexpected problem. The description can be open while the record of which machine or task it concerns stays exact.

FIGURE 2 · BUILD IN THIS ORDER

Entity → action → event.

IDENTIFY

Entity

The exact thing and its current state.

A work item, room, request, order or software version.

CONSTRAIN

Action

What may be done, and what must be true first.

Who may act, where, when and after which checks.

PRESERVE

Event

What happened, with its evidence.

Before → after; who acted, when, with whose permission and what was seen.

Where it belongs	The exact thing	What may change	Who may act
Business / task	Item / version	Stage / next step	Person or tool / permission

Identify the thing, define the allowed action, then record what happened. The words change between businesses. An attempted action and a checked result are still different facts.

05 · THE ARCHITECTURE

You choose the goal. The system carries the work through.

Suppose you want a piece of work delivered well. Choosing that goal is one job. Knowing the latest facts is another. Deciding the next step, checking permission, doing the task and checking the result are different jobs too.

A business brain connects them. People set the purpose and the limits. The system keeps the relevant facts available. AI uses them to work out a next step within those limits. Controls check that the step is allowed. A person, software service or device does the work. A check against reality tells the system what happened.

The distinction matters because a good explanation can still lead to a bad action. “This customer seems trustworthy” does not give permission to exceed a credit limit. “This is probably the test site” does not make it safe to change the live site.

A prompt can remind AI of a rule. A control at the point of action can enforce it, even when AI gets it wrong. The system needs both: reasoning that helps choose a step and limits that do not rely on the reasoning always being right.

Some work needs no AI at all. Adding numbers, checking a spending limit and preventing the same payment twice are jobs ordinary software can do. AI helps where the task needs interpretation, planning or an explanation. Each part should do the work it is suited to.

The point of all these parts is human: the owner can express what matters without repeating it to everyone; the worker can see what this task needs without fearing an invisible test. If the result is wrong, both can look at the same trail instead of arguing over what one person meant and the other person heard.

FIGURE 3 · THE OPERATING LOOP

Each part has a clear job.

01	**Goal and limits** People choose the goal, what matters and what the system may decide.	Human direction
02	**Context** Know the exact work, where it stands, its rules and what happened before.	Current facts
03	**AI direction** Work out the next step within the agreed limits. Say what is uncertain.	Reasoning
04	**Permission check** Before acting, check the right thing, the approval and the required conditions.	Controls
05	**Execution** A person, software service or device does the clear task.	The action
06	**Evidence** Look at what happened. Check it against the intended result.	Result check

↳ **Feed the result back.** Update the facts and keep the history. If something does not match, pause and follow it up.

These are six jobs, not necessarily six separate apps. One product may do several of them. Their roles still need to stay clear.

The loop stays open until the result comes back. Sending an instruction shows that an instruction was sent. It does not show that the right thing happened.

When the result differs from the plan, that changes what the system knows. It may need to stop the next step, repair a failed part or correct an assumption. The record of the work grows with the work itself.

The person doing the task does not need to see all of this at once. They need the action, the reason that matters to them and a way to report a problem. The full structure supports the simple instruction.

06 · A WORKED EXAMPLE

Different work can follow the same loop.

Take one simple need: move the right thing to the right place, with permission, and check that it worked. The thing might be a software update, a finished physical job, a file, a room ready for use or a clinic document.

First, find the exact work and the facts around it. Establish which moves are allowed. Choose the next step and give it to the right person or tool. Then check the result. COA names these steps **locate, constrain, direct, execute, prove**.

The examples below show the same loop in five settings. Each has three possible paths: the work may go ahead, a mismatch stops it, or the action has happened but its result still needs checking. These are teaching examples, not full procedures or measured results.

FIGURE 4 · THE SAME LOOP IN FIVE SETTINGS

Different facts.
The same questions.

Choose an example. See when work can go ahead, when it must stop and when it still needs proof.

Release a software change

Find it & check the rules

Right version + right system + checks passed

→

Choose the step & do it

Publish the approved change

→

Check the result

The intended task works on the live service

All checks pass → act, then check

The version, destination, approval and tests all match. Publish that version, then try the real task on the live service.

Something does not match → stop here

The change is about to reach the live service, but only a test release was approved. Block it and get the right approval.

Proof is missing → the task stays open

The publishing tool says success, but nobody has checked that the task works for a user. Keep that check open and investigate.

Complete a physical job

Find it & check the rules

Right item + latest requirements + ready tools

Choose the step & do it

Do the approved task

Check the result

Inspect the result on the actual item

All checks pass → act, then check

The item, requirements, tools and person's permission match the task. Do the work and inspect the result.

Something does not match → stop here

The item or the drawing is different from the one in the task. Stop this step and find out which is correct.

Proof is missing → the task stays open

The work is marked done, but the required inspection is missing. Record what was done and keep the item waiting for that check.

Deliver a digital service

Find it & check the rules	Choose the step & do it	Check the result
Latest brief + agreed work + approved version	Send it to the approved person	Check it arrived and meets the agreed requirements

All checks pass → act, then check

The approved version meets the brief and may be sent to this person. Send it, then check delivery and any required acceptance.

Something does not match → stop here

The version or recipient differs from the approval. Stop the send and resolve the mismatch.

Proof is missing → the task stays open

The delivery service accepted the file, but arrival or required acceptance is not confirmed. Follow up before closing the promise.

Release a room for use

Find it & check the rules

Right room + completed tasks + required checks

Choose the step & do it

Check the room is ready

Check the result

The room itself meets the readiness checks

All checks pass → act, then check

The room, completed tasks and permission to inspect all match. Check the room and record what is actually there.

Something does not match → stop here

A fault is still open, so the room cannot be marked ready. Send the fault for repair and keep the room waiting.

Proof is missing → the task stays open

Cleaning is marked done, but the required room check is missing. Keep the room waiting for that check.

Route a clinic document

Find it & check the rules	Choose the step & do it	Check the result
Right journey + right document + required sign-off	Send it to the approved next station	The right document is present at the right station

All checks pass → act, then check

The journey, document version and required professional sign-off match. Send the document to the approved next station.

Something does not match → stop here

The document belongs to another journey or lacks a required sign-off. Stop this step and ask the authorised person to resolve it.

Proof is missing → the task stays open

The software says it printed or sent the document, but the station has not confirmed the right copy. Check that gap. Clinical decisions remain with qualified people.

These examples explain the idea. Real controls must cover every route that can make the change. The final check must look at the result itself.

The person doing the task receives what they need for that step. An AI agent needs the same clarity: what to change, where to change it, what it is allowed to do and how to check the result. Both need a way to report a mismatch. A person may also supply a physical observation or skilled judgement that the software cannot provide.

Now suppose the system does not receive a reply. Should it try again? If the first attempt already charged the customer or sent the file, a blind repeat creates another problem. Each intended action needs its own identity, so a retry can check what has already happened.

Some attempts only partly succeed. A file may arrive while the receipt check fails. A task may finish while the record fails to save. Recovery has to begin from the real state of the work, not from the assumption that nothing happened.

“The agent finished” and “the worker said it was done” are both reports. The system still needs the evidence required for that job. It keeps the task open, follows up and checks again until it can state

what is known.

07 · ACROSS THE BUSINESS

One person's promise becomes another person's work.

A business starts making a promise before an order arrives. Marketing tells people what to expect. Sales agrees the details. Purchasing and staffing make the work possible. The team delivers it. Finance checks what is owed and what has been paid.

These are often treated as separate departments. The customer experiences one promise.

The customer is also a person carrying an expectation. They may have arranged their day, spent their money or trusted the business with something important. A broken handoff between departments reaches them as a broken promise, even if every team meant well.

Sales can agree a date the team cannot meet. Purchasing can get a lower price and fill a store with stock the business cannot use. Finance can pay an invoice while the delivery is still disputed. Each department may think it has done its job while the business has failed to keep the promise.

The brain needs to carry the connection between them. Before a promise is made, can the business keep it? Before money is paid, what was actually received? If one part changes, which other work is affected?

This does not require every department to see everyone else's records. Sales may need to know that capacity is available without seeing private staff details. A useful answer can cross a boundary while the records and permission to change them stay with the people responsible for them.

FIGURE 5 · ONE CONNECTED BUSINESS

Follow the promise through the business.

01

Attract

Make a truthful offer

02

Commit

Check terms and capacity

03

Prepare

Have the people and tools ready

04

Deliver

Do the work and check it

05

Settle

Check what is owed and paid

06

Learn

Find what happened and improve it

The line makes the connection easier to read. Real work also happens in parallel, comes back, gets cancelled or changes. The system needs those paths too.

Marketing and sales: promise what can be delivered.

An offer needs current terms, the right audience and enough capacity to fulfil it. AI can help write the offer. The system must still check the approval and spending limits before it is published. The finished draft is one result; the actual message people see is another.

A quotation then connects that offer to a particular customer. Which product or service? At what price? By when? With which exceptions? A salesperson can work with valid options instead of trying to remember every rule. An unusual request becomes a clear decision for someone who can approve it.

Purchasing and stock: keep each movement clear.

A purchase begins with a need, stock on hand, expected demand, supplier terms and a budget. Placing the order, receiving the goods, approving the bill and paying it are separate events. One “done” should not stand for all four.

Stock also has a place, a condition and someone responsible for it. If it moves, the record should show where it came from and where it went. If the count is wrong, a correction should keep the earlier count and the reason for changing it. Otherwise the business loses the very trail that could explain the difference.

Daily work and service: give the next person a ready task.

A task should arrive with the things it depends on already checked. The right item is available, the earlier step has finished and the method is clear. If something is missing, the system should deal with that gap instead of quietly passing it to the next worker.

The same applies when a customer has a problem. Find the promise, establish what went wrong, choose an allowed remedy and check that the customer received it. Recording a complaint is the start of that work, not its end.

Finance: know what the money stands for.

A bill is not owed just because AI can explain it. A payment is not settled just because a service accepted the request. The system must connect the obligation, the approval, the transaction and the check against the account. Each answers a different question.

AI can bring those facts together and flag a mismatch. Spending limits still apply, and business judgement and legal duties remain with the people responsible for them.

People and management: organise work without losing sight of people.

A schedule can connect the work with people who are available and trained to do it. A new colleague can receive the context their role needs. Repeated gaps can show where training would help.

But a task record is not the whole person. Hiring, pay and discipline need care, fair judgement and a way for the person affected to challenge a mistake. More records do not automatically mean a better understanding of someone's contribution.

Management brings these parts together. It should receive the decisions that remain unresolved, with the facts needed to decide. Routine work should move within agreed limits. Collecting every small problem on the owner's screen would only move the old burden to a new place.

08 · FROM TOP TO BOTTOM

The goal at the top must reach the work on the ground.

The idea reaches across departments, but it also runs from the owner's purpose down to the smallest task. The task should serve a real goal. The goal should eventually be judged by what happens on the ground.

“Serve more customers” sounds clear. But at what cost to quality, profit or the people doing the work? Someone has to decide which trade-offs the business will accept. AI can help compare choices. A number going up does not make the choice right.

Those choices become rules the system can use. The rules guide which work happens next. People and tools do that work. Evidence comes back to show whether it served the goal.

Permission can be given at each level. It needs clear limits, so the system knows when it can move and when a person must decide. An approved routine should not need the same approval every time it runs.

FIGURE 6 · FROM PURPOSE TO EVIDENCE

Level	The question	Example
Purpose	What are we trying to achieve?	People responsible for the business choose its promise and the trade-offs they will accept.
Policy	What may the system decide?	The right people set the limits, approvals and checks needed to call work done.
Coordination	What should happen next?	The system uses current facts and earlier steps to choose what may happen next.
Execution	What do I do here?	A person, software service or device does the clear task.
Evidence	Did the intended result occur?	A suitable check looks at what happened and compares it with the goal.

Each level needs the right permission. A suggestion still needs approval where required. An approval still needs a result check.

The owner still owns the direction. What changes is the need to act as the middle person for every small task. They can decide what the business should be, approve the commitments that need them and answer the questions only they can answer.

The system should then carry the work through. It finds the context, tracks what is waiting and checks the result. If the owner still has to remember every pending task and chase every reply, the burden has

not moved.

A useful question to bring back is a real choice with its consequences clear. “We have these two options; this is what each means.” A pile of unfinished work asking for attention is something else.

09 · ACROSS INDUSTRIES

The questions travel. The answers belong to each business.

Every business has its own language. The same basic questions sit beneath much of the work. What are we dealing with? Where does it stand? What may happen next? Who can do it? How will we know it worked?

Those questions can travel from one industry to another. The answers cannot simply be copied. A room, a software release and a medical document each need different knowledge and different checks.

FIGURE 7 · WHAT TRAVELS, AND WHAT CHANGES

One structure. Different business knowledge.

WHAT STAYS THE SAME

Exact thing + current stage → allowed next step → recorded action + checked result

Current facts · clear permission · a way to stop · a history you can follow

PHYSICAL & TRADITIONAL

Objects and places

Which thing, where it is, who holds it, its condition and how to check it.

SOFTWARE

Versions and environments

What should work, what it needs, who may change it and what works live.

DIGITAL SERVICES

Briefs and commitments

What was promised, who it is for, what is approved and whether it arrived as agreed.

Inside each business, the same idea connects departments and levels of responsibility.

Marketing · sales · resources · operations · finance · people

Purpose → policy → coordination → execution → evidence

The structure connects the questions. Knowledge of the business supplies the answers. Records and permissions stay with the business and roles they belong to.

A hotel room

“Room ready” may mean the previous guest has left, cleaning is done, a fault is fixed, supplies are in place and someone has checked the room. These are separate facts. A housekeeper needs the correct room and task. The next booking needs a readiness answer based on the room as it is now. The guest will experience the room, not the status on a screen.

A workshop

A job may need a particular drawing, material batch, machine setting and inspection. If the drawing changes, the system should find the unfinished work affected by that change. The craftsperson still supplies the skill and can notice a strange vibration or poor fit. Clear coordination helps that expertise reach the right job. An instruction alone cannot make a faulty machine safe.

A professional service

A piece of client work needs the agreed brief, the right source material, the approved scope and a clear final version. The system should block an old draft from being sent as the approved work and record who approved it. AI may help create the work. Quality still needs a check suited to it. The existence of a file does not prove that the design is good or the argument is sound.

A clinic's paperwork

A person's appointment, identity, documents and movement between stations need to stay connected. If the identity is unclear, the affected step should stop. The system can help the right document reach the right station with the required sign-off. Clinical decisions remain with qualified professionals. Organising that journey is different from deciding how to treat someone.

A software release

A change needs the exact code version, the right test or live system, permission, expected behaviour and a way to undo the change. Tests show some things about the code. A deployment shows that a version reached a place. Opening the real service shows whether the intended task works there. Each check tells a different part of the story.

These examples show why the structure could travel. They do not prove that one build will work everywhere. Each business needs its own model of the work, reliable facts and a useful check of the result. The value of building that structure must also justify its cost.

10 · PERMISSION

Knowing what to do does not mean being allowed to do it.

AI may find a sensible next step without having permission to take it. It may also have access to a tool that can do more than it is allowed to do. The system has to handle both. A suggestion needs the right approval, and the tool needs limits that enforce it.

Suppose a purchase would prevent a shortage. The system can prepare the order and explain what delay would mean. If the amount is above its spending limit, urgency does not create permission. The right person must decide. Once the defined purchase is approved, the system should carry it through without asking the same question again.

The rule must cover every route to the action. A screen might block an unapproved payment while an import, an API or another agent can still make it. In that case, the screen is careful but the business is not protected. The check belongs where the action takes effect.

Permission also belongs to a particular situation. Approval for one version does not cover any later version. Approval to send a message to one group does not cover a wider audience. When the facts change, the system must check whether the earlier approval still applies.

That clarity matters to the person doing the work. They should be able to act on a clear approval without worrying that an invisible condition will later be used to blame them.

The person doing the work needs a way to stop too. “This is the wrong item.” “This is unsafe.” “I cannot check this.” A system should make those reports easy and pause or redirect the affected work. A screen should never make a person ignore what is in front of them.

The record should make responsibility clear: who set the rule, who approved the action, what the system chose and what the person saw. Clear steps should reduce guesswork without hiding who is answerable for the result.

11 · PROOF

The world gets the last word.

A screen says success. An agent says finished. A dashboard turns green. Each can be a useful signal, but what exactly does it prove?

A message can be accepted by a delivery service and still fail to reach the person. A printer can accept a job and produce no paper. Stock can be recorded on the wrong shelf. A room can be marked clean while a guest finds a problem.

To the customer still waiting, a green screen does not feel like success. The check has to meet the life outside the screen.

Before doing the task, decide what would count as a finished result. Then choose a check that can see that result. A small text change needs a different check from a payment or a task where someone could be hurt. More checking is not always better; the check has to suit the consequence.

It also needs a chance to catch what the first step missed. If an agent does the work and writes “verified” without looking at anything else, the extra word proves little. Two dashboards reading the same broken source can agree and still be wrong.

This is why COA asks for a witness that can fail differently from the action it checks. That might be a fresh read from the destination, a physical inspection or another suitable observation. The independence needs checking too. Two sensors may share the same fault; two screens may share the same data.

No evidence sees everything. A photo may miss a detail. A sensor may drift. A person may misunderstand what to inspect. The system should say what its evidence confirms and what remains unknown. That is more useful than a broad claim that nothing can be wrong.

THREE DIFFERENT CLAIMS

“The instruction was sent.”
“The action was tried.”
“The intended result was seen.”

Only the third checks the outcome. Each claim needs its own evidence.

If the check fails, the task stays open. The system finds the missing link, follows an allowed route to fix it and checks again. If a person’s decision or physical action is needed, it brings that clear step to them. It should not depend on the owner remembering to chase it later.

12 · LEARNING

A mistake should change what happens next time.

A mistake happens. Someone sends a reminder. A procedure gets longer. Everyone is asked to be more careful. The next person still has to find and remember the lesson.

An apology can matter to the person affected. But if the work stays the same, the next person may have to apologise for the same mistake.

There is a stronger way to learn. Find what allowed the mistake, then change that part of the work. If a task happened twice, prevent a second attempt from creating a second charge or movement. If two items were easy to confuse, make their identities clear. If “done” hid a missing result, improve the check.

Not every mistake needs another rule. A misleading label may need better words. A needlessly complex task may need fewer steps. Adding checks around a bad design can make it harder to use without fixing the cause.

AI can look through the trail for patterns and suggest a change. It should not quietly give itself more permission or change live rules on its own. A proposed change needs someone responsible for it, a test, a version and a controlled way to introduce it. There should also be a way back if it makes things worse.

FIGURE 8 · LEARNING THAT REACHES THE OPERATION

01 →

Observe

Keep what happened and the evidence.

02 →

Explain

Find what allowed the mistake.

03 →

Change

Propose a change with a clear scope.

04 →

Evaluate

Test that it helps and does not break other work.

05 →

Authorise

Get approval for changes that need it.

06 ↰

Recheck

Check what happens once the change is in use.

A lesson reaches the work when it changes what the system does or checks. A written reminder alone cannot prevent the same mistake.

The history matters here. What facts were known at the time? Which rule applied? What was decided, and what followed? Without that sequence, it is easy to judge an earlier decision using facts that only appeared later.

A successful repair should also leave the failure visible. Otherwise the business may think the original process worked and never learn why it needed rescuing.

More history gives the system more to learn from. It does not guarantee improvement. The improvement comes when a sound lesson changes the work and a later check shows that the change helped.

13 · WHAT ONE PERSON CAN DO

You can do more than you can hold in your head.

You may see a useful idea in one field and want to build it in another. The difficulty is often everything that has to be learned and connected before the idea can become real. Software, buying, service, money and delivery each bring their own details.

AI makes much of that knowledge easier to reach. A business brain can help keep it connected from one task to the next. The question becomes less about how much one person can remember and more about how much sound work they can direct.

There is another kind of freedom in that. You can care about many things without having to carry each unfinished detail in your head or turn every missed step into disappointment with someone close to the work. Your attention can go back to choosing, creating and being present with people.

Think of the same person opening a service, improving a stock process and building a piece of software. The work needs different expertise. A useful system finds that expertise, keeps the purpose clear, chooses suitable tools and checks each result. It also recognises when a skill or permission is missing.

This is the practical meaning of the ambition to become a “master of all trades.” The capability can be built around the person. They do not have to store every rule in their own memory before they can act across fields.

It can reduce dependence on one person who alone knows how an operation works. It does not remove the need for skilled people, suppliers, institutions or the people who build the tools. If the business stops when one AI provider is unavailable, the dependency has moved.

So its memory must be able to survive a change of tools. Keep the source material, decisions and evidence in forms that can be recovered. Have another way to carry out important work. A single chat or a single device should not hold the only copy of what the business needs to know.

The claim is not that one person knows everything. It is that one person may be able to direct a wider range of good work because the system holds more of the context and follows more of the work through. The measure is what gets done well, not how capable the arrangement sounds.

14 · PEOPLE

A clear step should leave room for a thinking person.

Clear instructions can remove a quiet burden from work: trying to guess what someone expects and worrying that an unstated rule will be used against you later. That can make unfamiliar work easier to enter and reduce blame for mistakes the system helped create.

For the person giving the work, it can change the feeling too. Instead of testing whether another human remembered everything they meant, they can ask whether the business made the expectation visible. A missed result still matters. It no longer has to begin as a verdict on somebody's character.

The same structure can become harsh if it treats every pause as a failure or every disagreement as disobedience. A person needs to understand the reason that matters, report a mismatch and challenge a wrong record. "The system decided" cannot be the final answer to an unfair result.

There should be room to learn as well. The first view can show one clear step. A person who wants to understand more should be able to see the process behind it. Experienced workers should be able to improve that process with what they notice.

The evidence should stay close to the work it is needed for. Checking that an order reached a customer is different from tracking every detail of a worker's day. Keep only the information the purpose calls for, control who can see it and decide how long it is needed. A task record should not quietly become a score of a person's worth.

The rewards have to fit too. If people are punished for stopping, they will hide uncertainty. If speed is praised while checking is treated as a delay, the records will begin to say "done" too early. Honest reports need to be a valid part of doing the job.

The person should not have to carry the whole operation in their head. They should still be able to think, notice and speak. Those are part of how the system stays connected to reality.

15 · COSTS AND LIMITS

The brain needs care too.

A business brain does not appear just because AI is available. Someone has to sort out the records, define the work, connect the tools and decide what a good result looks like. Those things need care as the business changes.

The effort is more likely to pay off where work repeats, mistakes matter or many people pass tasks to one another. A rare, harmless task may only need a clear instruction. Creative work may need freedom in how it is done, while still needing a clear brief and permission to release the result.

AI adds cost, delay and another service the business relies on. Use it where reasoning helps. A settled rule should not need an expensive new explanation every time it is applied. The full cost includes checking, maintenance, recovery and the people who support the system.

There are limits in the model too. A correct record can point to the wrong physical object. A rule that worked yesterday can be wrong today. Someone can misuse an allowed path. Several failures can happen together and defeat a check designed for only one.

An AI failure is not automatically the owner's fault. A model may reason badly, a provider may go down, a record may be false or someone may act dishonestly. The owner's responsibility is to decide what the system may do, give it sound ways to check, notice when it fails and repair the part within their control. Blaming yourself for every event would recreate the same burden in another form.

So "the wrong move cannot be made" needs a clear scope. It can mean that a specified action is blocked under known conditions, through every route that can take it. It cannot mean that nothing harmful can happen. A rule can be enforced perfectly and still be a bad rule.

The business also needs a plan for a lost connection, an unavailable model or a failed service. Some work should pause. Other work may continue through an approved local process. Either way, keep enough identity and history to bring the records together later. Work that happens outside the trail can leave the restored system with the wrong picture.

The method is intended for every business layer. That is the reach of the proposal. Whether a particular build works well in a particular setting has to be shown. Naming the limits makes that question possible to test.

16 · PUTTING IT TO WORK

Start with one task, and follow it all the way through.

You do not have to rebuild a whole business to begin. Choose one piece of work with a clear start and a result you can check. It might be receiving a delivery, releasing a room, sending a quotation or publishing a software change.

Watch how it really happens. Include the side messages, the unusual cases and the person who quietly fixes things. The written process may leave out much of the work that keeps it running.

Ask where someone last said "I thought you knew" or "I thought that was done." Follow that moment without looking for someone to blame. It often reveals the exact expectation the system needs to carry.

Then identify the things involved and the stages they pass through. Define the allowed actions, the conditions for each one and who can approve them. Decide how a case outside those rules will be handled.

Build the way of using it around the person's real work. Someone with occupied hands may need to speak or scan. Someone reviewing a document may need a laptop. A software agent needs a precise tool. The structure should fit the task without asking people to build their day around it.

Connect each action to its check and its recovery path. What if the action happens twice? What if only half of it succeeds? What if a fact is missing? These are part of the first complete loop, not details to add after rollout.

Try the system within a limited scope. It can first suggest steps while people continue to do the work, so its choices can be compared with reality. Then allow it to act within agreed limits, with suitable oversight, and measure what happens.

Expand when the earlier loop works and the next connection is understood. Keep the rules, versions and history so the business can rebuild or change tools. A set of good screens is useful only if the work between them holds together.

17 · TESTING THE CLAIM

A clear example explains an idea. A fair test tells us whether it works.

The examples here make the proposal easier to see. To show that it helps, we need to compare results. We also need to ask which part helped. Clear records and better checks may improve a business even before AI is added.

A useful study would compare the arrangements below on similar work. This is a proposed study design. No results from it are claimed in this thesis.

FIGURE 9 · FIND OUT WHICH PART HELPS

A	B	C	D
Current practice	**Structured workflow**	**AI assistance**	**Full COA loop**
People use the process already in place.	Make the work, rules and checks clear, without AI choosing steps.	Add AI advice to the existing process without building the full system.	Connect context, AI, enforced rules, action and result checks.

This is a plan for a comparison, not a report of results. Compare similar work and account for differences in training, task difficulty and the quality of each build.

Five claims can then be tested.

1. **Making hidden knowledge visible should reduce avoidable mistakes.** Compare similar tasks. Count errors and examine whether missing or unclear context

caused them. Finishing faster is not enough if more work has to be repaired later.

2. **A rule that blocks an action should work better than a reminder alone.** Try the forbidden action through each real route: the screen, connected tools and background jobs. If any route still allows it, the claimed protection is incomplete.

3. **Checking the outcome should reduce false claims of completion.** Compare what the system calls done with what actually happened. Count missed failures, false alarms, time spent checking and time needed to recover.

4. **AI should add value where the work needs interpretation.** Compare the full loop with a good process that uses no AI. Measure quality, cost, response time, whether it asks for help at the right time and how it handles a change in the situation.

5. **Clear expectations may reduce avoidable blame.** Ask owners and workers about unclear instructions, repeated checking, trust and feelings of blame before and after a change. Compare their answers with observed work, because fewer complaints alone could mean people feel less safe speaking up. This thesis does not claim that any reduction has already been measured.

The people doing the work belong in the study too. Do they understand the next step? Is the work easier to manage? Can they stop a bad instruction and get it corrected? Are they learning? Does their trust match how reliable the system really is?

Where possible, assign comparable tasks or teams to the different arrangements and decide the measures before seeing the results. When that is not practical, a staged introduction can still teach us something. But changes in staffing, demand, season or experience may also explain a change in performance.

Rare failures need their own attention. An average can look good while hiding a small number of costly mistakes. A faster process that creates hidden rework or takes away people's ability to intervene has not fully succeeded.

The builder's own business is a useful place to find problems and test an early design. The builder also knows more about that setting than most later users will. Other people need to repeat the work in other

settings before the result can support a wider claim.

A result against the thesis is useful too. If a simpler process works just as well for less money, use the simpler process. If AI prevents one kind of error and creates another, show both. The argument becomes stronger by being open to a fair test, not by hiding what did not work.

18 · ORIGINS AND RELATED WORK

The contribution is how the parts are connected.

COA grew from Abdul Nihaf's working doctrine and the design of Nihaf Brain. The question running through that work is how a person can give an objective in ordinary language and have a system carry it through with the right context, permission and proof. Working documents from 2026 develop that idea, including the principle of zero hidden expectation.

Visa Medical is part of that design history. In the design work, one recurring problem was keeping the same journey connected across stations, documents and instruments. It also highlighted a useful distinction: software saying a document printed is different from the right document being present at the right station. Clinical judgement remains with qualified professionals.

These are sources for how the idea developed. They do not amount to a controlled study, or prove that every proposed physical layer has been checked. This publication also does not claim to have audited the full execution trail. A formal case study would need a defined set of records, a clear reason for including each case, and failures as well as successes.

Nihaf Brain adds the idea of continuity around a person. The person gives the purpose. The system finds the context, follows the work and brings back evidence or a choice only that person can make. Each business keeps its own records and authority. The method can travel without mixing customers, money, credentials or permissions between businesses.

The idea also sits beside established research. In *Cognition in the Wild*, Edwin Hutchins studies how thinking is spread across people, tools and their surroundings. This helps frame the idea that the knowledge needed for work can live beyond one person's memory. It does not prove that COA works. [1]

Research on business artifacts starts with the things a business handles and the stages those things pass through. This is close to COA's use of clear identities, allowed changes and records of what happened. Those foundations already have a history. Giving the arrangement a name does not make every part of it new. [2]

Lisanne Bainbridge's *Ironies of Automation* describes difficulties that automation can leave for people. Removing routine tasks can leave a person with only the hardest cases, without enough practice to

handle them. This matters to COA's exception path. People still need understanding, training and a real way to intervene. [3]

A review by Vaccaro, Almaatouq and Malone found that, on average across its included experiments, human–AI combinations did better than humans alone but worse than whichever performed better alone: the human or the AI. Combining them is not an automatic gain. The task and the division of work matter. [4]

A study of generative AI in customer support found productivity gains in that setting, with different effects across workers. It shows that useful gains can occur. It does not show that a whole business can run on its own or that expertise is no longer needed. [5]

COA's proposed contribution is the way the parts are joined: context, AI reasoning within agreed limits, enforced permission, a clear task, evidence of the result and tested changes to the system. The proposal applies this arrangement across departments and from the owner's goal to the work on the ground. Its value and originality need to be judged against other research and real implementations.

The thesis follows a conceptual and design approach: explain the problem, define what would be built, show how it would work, name the assumptions and make the claims testable. Research on design science and conceptual articles informs that structure. [6] [7]

In an earlier essay, Abdul Nihaf asked what AI might give back to a person whose brain has spent years keeping watch so their heart can stay safe. COA asks what that thought would look like where people spend so much of their lives: in the work they do together.

There is a small, painful sentence behind much of this work: “I thought they would know.”

Put that knowledge into the work. Then the person giving the task does not have to ask someone else to read their mind. The person doing it does not have to live under an invisible test. When something goes wrong, they can face the problem together.

Imagine that happening across a business, then across many businesses. Owners could build ambitious things without guarding every detail alone. Workers could bring their skill and care without fearing a rule nobody told them. Customers could trust promises because the work behind those promises is connected and checked.

That is more than a way to finish tasks. It is a chance to spend less of human life second-guessing, blaming and bracing for disappointment. AI cannot make people perfect, and it cannot make a heart peaceful by itself. But if it helps carry the watching that work has placed on our minds, we may have more room to trust, to be kind and to be present with one another. That is the kind of happiness this thesis hopes to make possible.

SOURCES & PUBLICATION

References

1. Hutchins, E. (1995). *Cognition in the Wild.* MIT Press. Publisher record ↗

2. Vaculín, R., Hull, R., Heath, T., Cochran, C., Nigam, A. & Sukaviriya, P. (2011). *Declarative Business Artifact Centric Modeling of Decision and Knowledge Intensive Business Processes.* EDOC 2011 conference paper. DOI ↗

3. Bainbridge, L. (1983). *Ironies of Automation.* Automatica, 19(6), 775–779. DOI ↗

4. Vaccaro, M., Almaatouq, A. & Malone, T. (2024). *When combinations of humans and AI are useful: A systematic review and meta-analysis.* Nature Human Behaviour, 8, 2293–2303. DOI ↗

5. Brynjolfsson, E., Li, D. & Raymond, L. (2025). *Generative AI at Work.* The Quarterly Journal of Economics, 140(2), 889–942. Published article ↗

6. Hevner, A., March, S., Park, J. & Ram, S. (2004). *Design Science in Information Systems Research.* MIS Quarterly, 28(1), 75–105. Journal record ↗

7. Jaakkola, E. (2020). *Designing conceptual articles: four approaches.* AMS Review, 10, 18–26. DOI ↗

Author's source material

This thesis draws on Abdul Nihaf's *Coordinate Operations Architecture doctrine* and the August 2026 working texts *COA Core Thesis* and *COA Zero-Expectation Thesis*. These are author-developed source materials, not peer-reviewed evidence. Private operational records are not reproduced here.

Publication note

Abdul Nihaf · The Quarter Engineer · First published 2 October 2026 · Revised 3 October 2026 · Version 1.3.

This is an independently published conceptual thesis. It has not undergone journal peer review, been awarded a university degree, or received formal accreditation. The website publication establishes a public version of the argument; those other forms of assessment require their own processes.

The operating doctrine and direction originate with Abdul Nihaf. AI assisted with research organisation, drafting, explanatory design, development and visual explanation. Version 1.1 introduced diagrams that explain the logic across five kinds of business. Version 1.2 rewrote the explanations in plainer language, simplified the diagram labels and repaired word spacing on narrow screens. Version 1.3 makes the human cost of hidden expectations central to the argument and adds a diagram and a testable claim about blame. The diagrams and interactive examples are authored explanatory models, not experimental results. Responsibility for the published work remains human.

The visual presentation takes inspiration from Distill's authoring guidance and Nature's figure guidance: figures sit beside the argument they explain, with legible labels and captions that identify their limits.

Suggested citation

Nihaf, A. (2026). *A brain for every business: Coordinate Operations Architecture—a thesis on context, AI direction and accountable execution.* Version 1.3. The Quarter Engineer. https://thequarterengineer.com/thesis/