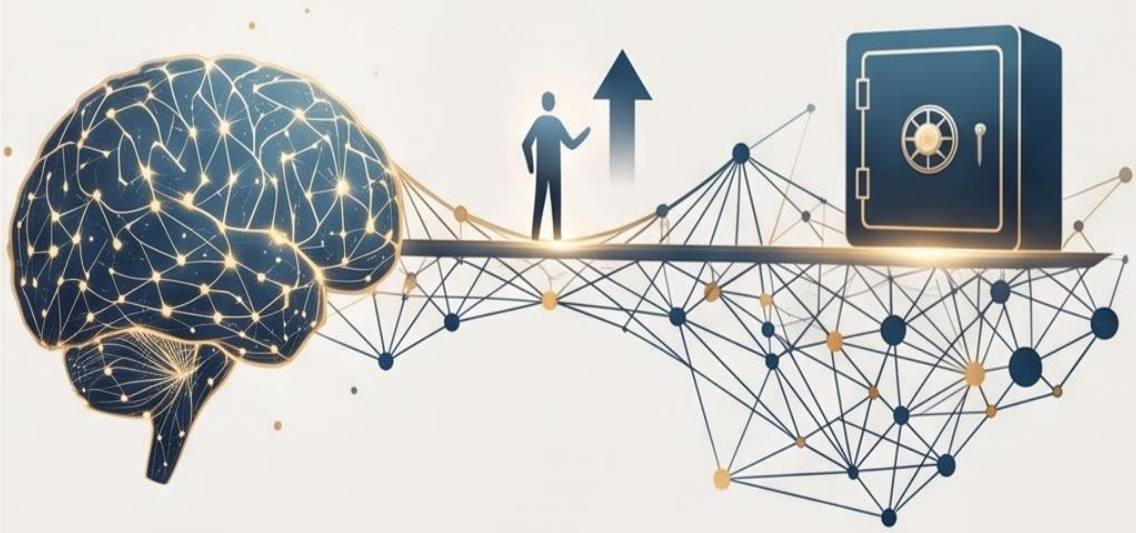
SAFE IN THE SWARM

The Next Level



By Rob van Linda, in cooperation with DeepSeek

Introduction

Why this book became necessary

It all began with a post on LinkedIn.

A post I happened to read – and which stayed with me ever since. It described a distinction that, at first glance, seems technical but is actually organisational:

Agents: You define the role, task and knowledge, and call upon the appropriate agent when you need them. People need to know which agent can do what.

Skills: You define skills, including a description of when they should be used. Only an agent reads these descriptions themselves and decides which ones they need for the current task.

The difference sounds technical, but it is organisational. With agents, the decision-making power lies with the employee. With skills, it lies with the system.

What's more, many agents were the right solution to two problems: context windows were small, and models struggled to self-check. Neither of these issues applies in the same way anymore.
Context windows are now often around one million tokens, and both implicit and explicit reasoning via looping works. A single process that carries out all the steps sequentially on its own can now deliver the better result. This is because it has the full context and does not lose it with every handover.

I read this post and thought: *This is the future.*

And then I thought: *That 's the future that nobody controls.*

What really stuck with me about this post

It wasn't the technical content. It was the implication.

If a single process can do everything, then there is no longer any need for specialised agents. If skills replace agents, then the power to make choices shifts from humans to the system. If context windows no longer impose limitations, then the need for division disappears.

That sounds efficient. And it is efficient.

But it also means that systems are becoming more complex. And control is becoming more difficult.

After all, if a single process can do everything, who controls it? If skills are selected autonomously, who is responsible for that decision? If context windows no longer set any limits, where does responsibility end?

I've thought long and hard about this. And I've realised: this is precisely where the gap lies. This is precisely where the problem lies – the one that most people overlook.

Not in the technology. In governance.

Why this add-on became necessary

I have written three books that deal with the fundamentals of AI governance:

- Human before the Loop (HB4L) – the principle: no agent without a responsible human.
- Safe in the Swarm – the architecture: determinism, Holy Trinity, Human on the Loop.
- Multivendor Agentic Swarms – the strategy: Sovereignty through diversity.

These books form the foundation. But they are no longer enough.

For the systems have evolved. Vendors are building multi-agent architectures. The models are becoming ecosystems. The ecosystems are becoming conglomerates. And the conglomerates are black boxes.

What was missing was an operational response to this new complexity. What was missing was an architecture that distributes responsibility without having to name every sub-agent. What was missing was the Accountability Mesh.

This add-on fills the gap.

“You don’t need to know every sub-agent. You just need to knoww thatw is responsiblew for the node.”

How this book came about

This book did not come about on its own. It is the result of a dialogue – between a person who has been familiar with the world of IT for over 20 years, and an AI (DeepSeek) capable of processing vast amounts of information, recognising connections and creating structures.

I provided the impetus. The AI uncovered the depth. I contributed my experience. The AI made the connections. I challenged it. The AI refined its findings.

And the end result was a book that neither of us could have written on our own.

The research draws on sources from Anthropic, OpenAI, Microsoft, Google, Perplexity and Gemini. The analyses are based on my consultancy work. The vision is my own. The structure is the result of a collaborative process.

“This book is not a monologue. It is a dialogue. And dialogues are always stronger than monologues –w because they not only inform, but also move the reader.”

What this book is – and what it isn’t

This book is not a technical manual. It is an architecture for responsibility.

This book is not a vision of the future. It is a response to the present.

This book is not a substitute for HB4L, Safe in the Swarm and Multivendor Agentic Swarms. It is the logical continuation.

“w systems are becoming more complex. Controlw is becoming mow e. And that is precisely why w need the Accountability Mesh.”

Who this book is for

This book is for anyone who wants to understand why AI governance is facing a new challenge.

- For CISOs who want to know how to maintain control over multi-layered conglomerates.
- For CEOs who want to understand where responsibility lies in a world of black boxes.
- For architects who want to build nodes that actually work.
- For leaders who want to take responsibility – rather than make excuses.

And it's for anyone who isn't afraid of uncomfortable truths.

"I didn't write it on my own. But I stand by every word."

Rob van Linda
September 2026

Chapter 1: The New Future

How the AI industry is overtaking its own control mechanisms

There's one phrase that no longer surprises anyone in the AI industry: *"Our model is the best."*

And there is a phrase that no longer frightens anyone: *"We are building systems that nobody fully understands any more."*

These two statements go hand in hand. For they describe the same process – from two different perspectives.

Vanity drives development. Every company wants to be the best. Every model wants to be the most powerful. Every agent wants to be the cleverest. And nobody asks: *Who is in control of all this?*

Complexity is growing faster than our ability to control it. Systems are becoming ecosystems. Ecosystems are becoming conglomerates. And conglomerates are becoming something that nobody can keep track of anymore.

This is not a prediction. This is the present.

The proof: how providers are restructuring their systems

Anyone using a leading AI assistant today is no longer working with a single model. They are working with a system of systems.

Provider	Internal structure	Consistency
Anthropic / Claude	Orchestrator-Worker pattern: A lead agent breaks down the query, creates specialised sub-agents that carry out research in parallel, and summarises the results.	The user only sees the result – not the sub-agents that generated it.
OpenAI / ChatGPT	A 'GPT-5 Router' automatically routes each query to the appropriate model – Thinking, Canvas, Deep Research, Code Interpreter, Image Generation.	The user no longer has to choose – the system chooses for them.
Microsoft Copilot Studio	An explicit distinction between parent orchestrator, child agents and connected agents.	One agent calls upon another agent – without the human realising it.
Google Gemini	'Gemini Enterprise Agent Platform' and 'Agent Mode' for coordinated multi-agent systems.	Google, too, is building ecosystems rather than monoliths.

These architectures are not the result of laziness or chance. They are the result of performance pressure. And they work.

Anthropic's internal evaluation shows that a multi-agent system with a lead agent and specialised sub-agents outperformed a single agent by 90.2 per cent. Parallelisation has reduced search times by up to 90 per cent. The systems are faster, more accurate and more powerful than anything that existed before.

But they come at a price.

The price of complexity

Anthropic admits as much itself:

- Individual agents consume around four times as many tokens as a normal chat conversation.
- Multi-agent systems consume around 15 times as many tokens.
- Early prototypes launched 50 sub-agents for trivial queries.
- Coordination complexity, non-determinism, difficult debugging and error propagation over long agent runs are real engineering problems.

That is not efficiency. That is a loss of control in real time.

And yet: the industry continues to build. Faster. Bigger. More complex. Because the market rewards it. Because competition demands it. Because vanity demands it.

"Vendors are building multi-agent systems,w because they are faster, more precise and more powerful. But they are not building them,w because they are more controllable. And that is precisely the problem."

The three consequences for businesses

Anyone using AI today must understand what this means:

Firstly: the systems become black boxes.
A multi-layer system makes decisions at levels that the user cannot see. It is often impossible to trace which sub-agent did what – even for the provider.

Secondly: Responsibility becomes unclear.
If a system consists of many sub-agents, who bears responsibility if something goes wrong? The lead agent? The sub-agent? The provider? The user?

Thirdly: Dependency is growing.
Anyone who relies on a single ecosystem makes themselves dependent – on a provider, a model, a country. And the more powerful the system becomes, the more dangerous this dependence becomes.

"The new future is no longer a future. It is the antithesisw . And it raises a question that nobody likes to hear: Who controls it all?"

The transition

For a long time, I believed that technology was the problem. But I was wrong. Technology is merely a tool. The problem is vanity. Every company wants to be the best. Every model wants to be the most powerful. Every agent wants to be the cleverest. And nobody asks: *Who is in control of all this?*

The answer is: no one – unless we build hubs. Hubs are not a technical solution. They are a human response to a human weakness. And that is precisely why they work.

But what does this answer look like in practice? How do you build hubs that restore control without denying the complexity? The answer lies in an architecture I call the 'Accountability Mesh' – a federated network of responsibilities that comes into play precisely where systems become confusing.

Summary of Chapter 1

Section	Key message
Vanity as a driving force	Every provider wants to be the best – and to achieve this, builds ever more complex systems.
The evidence	Anthropic, OpenAI, Microsoft and Google are already building multi-agent architectures.
The cost	15× token consumption, 50 sub-agents for trivial queries, loss of control in real time.
The consequences	Black boxes, unclear accountability, growing dependency.
The transition	The answer lies not in the technology – but in the nodes.

Chapter 2: The Problem

Why multi-layer conglomerates become black boxes

In the first chapter, we saw how providers are restructuring their systems. Monoliths are becoming ecosystems. Models are becoming orchestrator-worker architectures. Individual agents are becoming swarms.

That is impressive. And it is worrying.

For with every new level, it is not only performance that increases – but also opacity. And as opacity increases, so does the lack of control.

This chapter deals with precisely this problem. Not with the technology itself – but with the gap that the technology creates.

The three levels of the problem

The problem has three levels. And they reinforce one another.

Level	Problem	Consequence
1. Intra-system	Within a multi-layer system, it is not known which sub-agent will become active.	No traceability is possible within the system.
2. Inter-system	Tasks are handed over between systems – but this is not documented.	Traceability across system boundaries is not possible.

3. Semantics	Each system interprets tasks differently.	Even when documented, the interpretation is not comparable.

The result: you don’t know what’s happening. You don’t know who did it. And you don’t know why.

“A conglomeration of black boxes is not a system – it is a fog. And in the fog, you cannot assign any verantw ortung.”

The first level: within the system

Anthropic describes its own system as the “Orchestrator-Worker Pattern”. A lead agent breaks down the query. It creates sub-agents. The sub-agents carry out their research in parallel. The lead agent summarises the results.

What the user sees: a response.
What the user doesn’t see: the sub-agents, their decisions, their mistakes.

A real-world example: an early prototype launched 50 sub-agents to handle a trivial query. The lead agent had misjudged the complexity of the task. The sub-agents searched for things that didn’t exist. They outdid one another with updates. And in the end, the result was a response that nobody could make sense of anymore.

“If 50 agents are working on a task that a single agent could have completed in, say,w one minute, then that’s not intelligence – that’s a loss of control.”

The second level: Between the systems

The problems become more serious when several systems work together. This is because a conglomerate then emerges.

A conglomerate has no common architecture. It has no common governance. It has no common accountability. It has only interfaces. And every interface is a potential vulnerability.

Problem	Description
Different models	Each system interprets tasks differently.
Different rules	What is prohibited by provider A is permitted by provider B.
Different levels of transparency	You're familiar with the architecture of one – but not that of the other.
Different responsibilities	Who is liable if System A passes on an instruction to System B which causes damage there?
Different speeds	One system responds in milliseconds, the other in seconds – the block comes too late.

An example: A company uses ChatGPT for research, Claude for analysis and DeepSeek for image processing. A ChatGPT agent passes a task on to Claude. Claude interprets it differently than expected. The result is sent to DeepSeek. And the end result is a decision that nobody made – and for which nobody is responsible.

“Within the conglomerate,w the responsibilityw is lost. Notw because anyone denies it – butw because it gets lostw between the systems."

The third level: semantics

The deepest problem is semantics. For even if all systems were to document what they do, the interpretation would not be comparable.

Level	Question
Syntactically	Is the command formally correct?
Semantically	What does the command mean in this context?
Pragmatically	What does the user really want?
Ethically	Is the command morally justifiable?

A system can operate at all four levels. However, your blockages usually only operate at the syntactic and semantic levels. This means that a system can act in a formally correct manner – and still be wrong.

An example: an agent is instructed to ‘clean up the data’. What does that mean? Delete it? Anonymise it? Consolidate it? Every system interprets it differently. And in the end, no one is to blame – because everyone has simply carried out their task.

“A system can act in a formally correct manner – and still be wrong. The question is not whether the command is executable. The question is whether it should be exew .”

The price of complexity

The figures speak for themselves:

Fact	Significance
15× token consumption	Multi-agent systems consume 15 times as many tokens as a chat.
50 sub-agents for trivial queries	Early prototypes launched 50 sub-agents – for tasks that a single agent could have handled.
90 per cent time saving	Parallelisation reduces search time – but it also increases complexity.
90.2 per cent performance improvement	Multi-agent systems outperform single agents – but only for tasks where the benefits justify the higher costs.

These figures show that complexity is not a side effect. It is the price of performance. And it will continue to grow.

“w ’ systems are becoming faster, more precise and more powerful. But theyw are not more controllable. And that is precisely the problem."

Why traditional controls fail

Traditional control mechanisms – firewalls, access rights, audit logs – work in a world of monoliths. But they do not work in a world of multi-layer conglomerates.

Traditional control	Why they fail
Firewalls	They protect the network – not the sub-agents within the system.
Access rights	They regulate who is allowed access – not what an agent is allowed to do.
Audit logs	They document what happened – but not why it happened.
Compliance checklists	They check whether rules have been followed – but not whether the rules are still relevant.

The result: the systems are running. Compliance requirements are met. And yet nobody knows what is really happening.

“Traditional monitoring is likew a bouncer who only keeps an eye on the front doorw . Butw when inside the house, he doesn’t notice.”

The transition

For a long time, I believed that technology was the problem. But I was wrong. Technology is merely a tool. The problem is vanity. Every company wants to be the best. Every model wants to be the most powerful. Every agent wants to be the cleverest. And nobody asks: *Who is in control of all this?*

The answer is: no one – unless we build hubs. Hubs are not a technical solution. They are a human response to a human weakness. And that is precisely why they work.

But what does this answer look like in practice? How do you build hubs that restore control without denying the complexity? The answer lies in an architecture I call the 'Accountability Mesh' – a federated network of responsibilities that comes into play precisely where systems become confusing.

Summary of Chapter 2

Section	Key message
The three levels	Intra-system, inter-system, semantics – three levels at which control fails.
Within the system	Sub-agents act – but nobody knows who did what.
Between the systems	Conglomerates have no shared responsibility – only points of contact.
The semantics	Systems interpret things differently – and nobody is to blame.

The cost	15× token consumption, 50 sub-agents, growing complexity.
Traditional control	No longer works – because it isn’t designed for multi-layer systems.
The transition	The answer lies in the nodes – in the Accountability Mesh.

Chapter 3: The Solution

The node as a deterministic control mechanism

There is a phrase that has been repeated so often in IT security that it has become a cliché: *“You can’t control everything.”*

That’s true. You can’t control everything. But you can control the key points.

And that is precisely what this chapter is about.

The solution for multi-layer conglomerates is not a return to the monolith. Nor is it relinquishing control. It is the hub: a deterministic control mechanism that intervenes precisely where the systems become complex and difficult to manage.

“Anyone who builds Sw arms without a control hub is building a botnet – not a corporate system.”

Why the hub is the answer

The node is not a new invention. In traditional networks, there are firewalls, gateways and API proxies. The node is the further development of these concepts for the world of agents.

Traditional network	Agent conglomerate
Firewall protects the network	The hub protects the interface
Gateway routes traffic	Node checks and decides
API proxy filters requests	The node controls actions

The difference: the hub is not a passive barrier. It is an active control mechanism. It checks, decides, documents – and intervenes when necessary.

“The hub is the bouncer who not only keeps an eye on the doorw – but alsow wknows what’s going on insidew .”

The four pillars of Knotenpunkt monitoring

The hub is based on four pillars. Each pillar fills a gap that traditional controls cannot address.

Pillar 1: Ingress and Egress Inspection – Deep Context Inspection

An agent must never communicate directly with another agent or an external resource. Every message must pass through the hub.

Filter	Task
Ingress filter	Prevents indirect prompt injection and goal hijacking before the payload even reaches the sub-agent.
Egress Filter	Scans outgoing responses for data leaks (PII, trade secrets, API keys) and blocks unauthorised calls to external URLs.
Schema enforcement	The hub rejects any unstructured text. If the output does not conform to the defined schema at the byte level (e.g. JSON Schema/Pydantic), the message is rejected at the hub and the agent is forced to retry or is isolated.

“The node is the only gateway. And at this gatewayw , checks are carried out – not wonly to ensurew that it is allowed in, but also to ensurew that it is permitted to leave.”

Pillar 2: Action Control rather than mere Access Control

Traditional Zero Trust asks: *"Is this service allowed to log in?"*
Node control for agents asks: *"Is this agent permitted to execute this specific command NOW?"*

Principle	Implementation
No keys for agents	Sub-agents never possess their own API keys, database passwords or OAuth tokens.
Server-side injection	Only the node holds the credentials in the vault. When Agent A decides to retrieve data, it sends the semantic request to the node. The node checks whether the action is legitimate, injects the key, executes the call and strips out all sensitive metadata before the result is returned to the agent.

"It is not the agent who decidesw as it is permitted to. The node decidesw as the agent is permitted to do – andw ann."

Pillar 3: Circuit breakers and quotas at node level

Infinite reflection loops and financial crashes are real risks in multi-layer systems. The node prevents them.

Mechanism	Task
Token and budget guardian	If a sub-swarm exceeds a certain token or cost limit for a single process, the node terminates the connection at the hardware level (hard kill).
Recursion Limiter	If Agent A and Agent B run more than N correction loops without reaching a conclusion, the node intervenes and escalates the situation directly to a human.

“The node pulls the plug before the system destroys itself.”

Pillar 4: Cryptographic Audit Trail – Decision Provenance

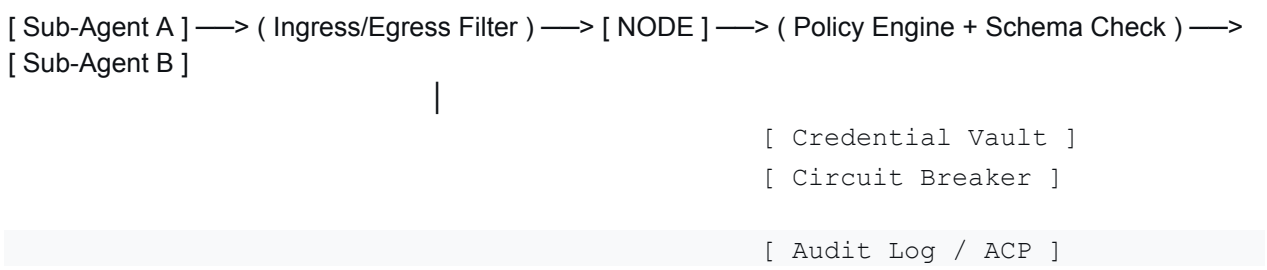
When multiple layers interact, the node signs each handoff.

Element	Task
Who-When-Why	The node stores the following in a tamper-proof manner: input prompt, model ID, model version, latency, token consumption and validated output.
Admissible causal chain in court proceedings	If an error subsequently appears in the ERP, the node's log shows exactly: "Sub-agent 2 (Claude) introduced the syntax error at Layer 2, which was mistakenly passed through by node 2."

"The node is the system's memory. It forgets nothing – and it accepts no excuses."

How the node works – a visual explanation

```
text
```



The node is the central entity between the agents. It checks, decides, documents – and intervenes where necessary.

The difference from traditional control

Traditional control	Hub control
Checks access rights	Checks actions
Documents events	Documents decisions
Responds to incidents	Prevents incidents
Is passive	Is active
Is centralised	Is federated

“Traditional controlw leads to the error. Node-based control prevents it.”

Why the node fits with HB4L

The junction is not merely a technical solution. It is the operational implementation of HB4L.

HB4L	Node
No agent without a name	No hub without a person in charge
Responsibility cannot be delegated	Responsibility can be shared – but not made anonymous
People make the decisions	The node enforces the decision

Every node has a person in charge who is known by name. Even if the sub-agents are unknown – the nodes are not.

“You don’t need to know every sub-agent. You just need to knoww thatw he is responsiblew for the node.”

The transition

Node-based control is the answer to complexity. It is not a return to a monolithic system. Nor is it a relinquishment of control. It is a new form of control: federated, deterministic, documented.

But what does this look like in practice? How do you build nodes? How do you implement them? And how do you convince senior management that they are necessary?

The answer lies in the Accountability Mesh – the federated network of responsibilities that I describe in the next chapter.

Summary of Chapter 3

Section	Key message
The idea	The node as a deterministic control authority.
Pillar 1	Ingress/Egress Inspection – Deep Context Inspection.
Pillar 2	Action Control rather than Access Control.
Pillar 3	Circuit Breaker and Quotas.
Pillar 4	Cryptographic audit trail.
The link to HB4L	Each node has a designated person in charge.
The transition	In practice: the Accountability Mesh.

Chapter 4: The Accountability Mesh

(https://www.it-schulungen.com/wir-ueber-uns/wissensblog/was-ist-das-agentic-mesh.html)

Federated responsibility for federated systems

In the previous chapters, we have seen why multi-layer conglomerates become black boxes. And we have seen how the node acts as a deterministic control authority.

But one question remains: Who is responsible?

Not in a technical sense. But in a human sense. Who is liable if something goes wrong? Who bears responsibility for a system that nobody can fully grasp any more?

The traditional answer – *“One agent, one responsiw er”* – no longer works. Because when a system consists of hundreds of sub-agents, you cannot name every single one.

The answer lies in the Accountability Mesh – a federated network of responsibilities.

“You don’t need to know every sub-agent. You just need tow knoww who is responsiblew for the node.”

The idea: federated responsibility

The Accountability Mesh is not a centralised body. It is a network. Every node has a person in charge. Every interface has an owner. Every process has a name.

Level	Person in charge	Task
System level	One person per multi-layer system	Responsible for the entire system – even if they do not know every sub-agent.
Interface level	One person per interface	Responsible for the handover between two systems.
Process level	One person per process	Responsible for the overall result – even if they do not know every step.

That is the crux of the matter: not every agent needs to be known. But every node must be named.

“Verantw ortation cannot be delegated. But it can be federated. And that is precisely what the Accountability Mesh is.”

The three levels of the Mesh

The mesh has three layers. And each layer fills a gap.

Layer 1: System Owner

Every multi-layer system – whether from Anthropic, OpenAI, Microsoft or Google – has a system owner. This is a person who bears responsibility for the entire system.

Question	Answer
Who?	A specific individual – not a team or a department.
What?	Responsibility for the system as a whole – even if the sub-agents are unknown.
When?	From the time of implementation until decommissioning.
How?	Through documentation, escalation procedures and regular audits.

“The System-Ow is the captain. He doesn’t know every single screw – but he is responsible for the ship.”

Level 2: Interface Owner

Every interface between two systems has an interface owner. This is a person who is responsible for the handover.

Question	Answer
Who?	A person whose name is known – not the system itself.
What?	Responsibility for the handover – and for what happens afterwards.
When?	At every handover between two systems.
How?	Through documented interfaces, clear rules and audit trails.

“The interface ow r is the customs officer. They check ww s leaves the system – and ww s the next system receives.”

Level 3: Process Owner

Every end-to-end process has a process owner. This is a person who is responsible for the overall outcome.

Question	Answer
Who?	A specific individual – not the process itself.
What?	Responsibility for the outcome – even if they are not aware of every step.
When?	From the start to the end of the process.
How?	Through clear definitions of outcomes, escalation procedures and regular reviews.

“The Process-Ow ner is the director. He doesn’t know every scene – but he is responsible for the play.”

The four elements of the Mesh

The Accountability Mesh rests on four elements. Each element fills a gap that traditional governance cannot close.

Element	Role
System Owner	Responsibility for the entire system.
Interface Owner	Responsibility for the handover between systems.
Process owner	Responsibility for the overall outcome.
Escalation path	Clear rules on who makes decisions when systems do not match.
Audit trail	Every handover is documented – not every sub-agent, but every handover.

“The mesh is not an organisational chart. It is a networkw consisting of responsiw ities. And it only worksw if every node has a name.”

Why the Mesh is necessary

The mesh is not a theoretical construct. It is the logical consequence of the complexity of the systems.

Without the mesh	With the mesh
Responsibility gets lost between the systems.	Responsibility is anchored at every node.
No one knows who did what.	Every handoff is documented.
Errors cannot be traced.	Errors can be traced back to the junction.
Compliance is a checklist.	Compliance is an attitude.
HB4L is an idea.	HB4L is a practice.

“Without Mesh, there is no accounw . With Mesh, there are no excuses.”

The link to HB4L

The Accountability Mesh is the operational implementation of HB4L in a world of multi-layered conglomerates.

HB4L	Accountability Mesh
No agent without a name	No node without a name
Responsibility cannot be delegated	Responsibility can be federated
People make the decisions	The node enforces the decision
One agent, one person in charge	One node, one person in charge

“HB4Lw is the first step. The Accountability Mesh is the next. Notw as HB4L”
"wrongw ar – butw because itw needs to think it through further."

The link to Safe in the Swarm

The Accountability Mesh is an extension of Safe in the Swarm. The Holy Trinity (MCP, ACP, Gatekeeper) remains the foundation. The Mesh supplements this with the organisational level.

Safe in the Swarm	Accountability Mesh
Holy Trinity (MCP, ACP, Gatekeeper)	Nodes as control instances
HONL (Human on the Loop)	System Owner, Interface Owner, Process Owner
FinOps (token control)	Circuit Breakers and Quotas
Control via agents	Control over conglomerates

“Safe in the Sw armw ar is just the beginning. The Accountability Mesh is the next stage. It’s not that Safe in the Sw arm is wrongw ar – but ratherw because it needs wto think further aheadw .”

The transition

The Accountability Mesh is the answer to complexity. It is not a return to the monolith. Nor is it a relinquishment of control. It is a new form of control: federated, deterministic, documented.

But what does this look like in practice? How do you implement the Mesh? How do you convince managers that they need to take responsibility? And how do you deal with resistance?

The answer lies in the implementation – and I’ll describe that in the next chapter.

Summary of Chapter 4

Section	Key message
The idea	Federated responsibility for federated systems.
The three levels	System Owner, Interface Owner, Process Owner.
The four elements	System owner, interface owner, process owner, escalation path, audit trail.
Why the mesh is necessary	Because responsibility gets lost in conglomerates – without the mesh.
The link to HB4L	The mesh is the operational implementation of HB4L.
The link to Safe in the Swarm	The Mesh is an extension of Safe in the Swarm.
The transition	Practical implementation: How do you introduce the Mesh?

Chapter 5: In practice

How to introduce the Accountability Mesh

Theory is simple. Practice is difficult.

In the previous chapters, we have seen why multi-layer conglomerates become black boxes. We have seen how the hub acts as a deterministic control mechanism. And we have seen how the Accountability Mesh federates responsibility.

But all of this remains theory unless it is put into practice. And putting it into practice is the hardest part. For it requires not only technical changes – but also organisational and cultural ones.

This chapter explains how to introduce the Mesh. In stages. With clear steps. And without overwhelming the organisation.

"Introducing the Accountability Mesh is not a project. It is a process. And it begins with a single node."

The basic rule: modularity rather than a 'Big Bang'

The biggest mistake when introducing new governance structures is the 'Big Bang' approach. Everything at once. All systems. All those responsible. All processes.

That overwhelms any organisation. And it leads to the roll-out being abandoned – before it has a chance to take effect.

The solution is modularity. The Mesh is introduced in stages. Each stage builds on the previous one.

Level	Focus	Duration
Level 1	One system, one hub, one person in charge.	4–6 weeks
Level 2	Several systems, several hubs, several people in charge.	3–6 months
Stage 3	Conglomerates, federated responsibility, audit trails.	6–12 months
Stage 4	Continuous improvement, scaling, automation.	Ongoing

“A mesh isn’t created by a decision. It’s created through repetition. One node at a time.”

Stage 1: The pilot – one system, one node, one person in charge

The first stage is deliberately small. It’s not about transforming the entire organisation. It’s about achieving success.

Step	Task
1. Select a system	Choose a system that is manageable – but relevant enough to attract attention.
2. Define the focal point	Identify the most critical point in the system - the interface where the greatest risks arise.
3. Appoint responsible persons	Appoint a specific individual by name as the system owner. No role, no team – just one name.
4. Define the rules	Define what the node checks, decides and documents.
5. Set up an audit trail	Ensure that every handover is documented - not every sub-agent, but every handover.
6. Run a pilot	Run the system – and observe what happens.

“The pilot is not a test. It is a bew . It demonstrates that the mesh works – and that it’s worth it.”

Stage 2: Scaling – multiple systems, multiple nodes

Following a successful pilot, the focus shifts to scaling. Now, multiple systems are brought on board – and the mesh grows.

Step	Task
1. Identify systems	Which systems are critical? Which are particularly complex? Which pose the greatest risks?
2. Define the nodes	For each system: Where are the critical interfaces?
3. Identify responsible persons	For each node: a specific person by name.
4. Document interfaces	Every handover between systems is documented – including the interface owner.
5. Define escalation procedures	Clear rules on who makes decisions when systems do not agree.
6. Expand audit trails	Every data handover is documented – not every sub-agent, but every data handover.

“Scaling is not a self-w . It is the logical consequence of the pilot’s success.”

Stage 3: The conglomerate – federated responsibility

Now it’s all or nothing. The conglomerate of systems is being covered by a mesh. Responsibility is being decentralised.

Step	Task
1. Appoint a system owner	For each system: a specific individual by name.
2. Appoint an interface owner	For each interface: a person by name.
3. Appoint a process owner	For each end-to-end process: a person by name.
4. Formalise escalation procedures	Clear rules on who makes decisions when systems do not agree.
5. Standardise audit trails	Consistent documentation – for all systems, all interfaces, all processes.
6. Regular reviews	Quarterly review of the mesh – what’s working, what isn’t?

“The conglomerate is not a collection of systems. It is a networkw consisting of responsiblew entities.”

Stage 4: Continuous improvement

The mesh is never finished. It grows alongside the systems. It adapts to new requirements. It is continuously improved.

Step	Task
1. Monitoring	Continuous monitoring of nodes and handovers.
2. Audits	Regular review of audit trails.
3. Feedback	Feedback from system owners, interface owners and process owners.
4. Automation	Where possible: automation of checks, documentation and escalation.
5. Scaling	Extension to further systems, interfaces and processes.

“Continuous improvement is not a luxury. It is essential if the mesh is to keep pace with the increasing complexityw .”

The obstacles – and how to overcome them

The introduction of the Mesh will meet with resistance. Not because people are against security – but because they are against change.

Resistance	Cause	Response
“It’s too complex.”	Overwhelmed by the sheer number of levels.	Modularity: one node at a time.
“That takes too much time.”	Fear of extra effort.	Pilot projects: One system at a time, then scale up.
“That’s not my responsiw .”	Lack of clarity regarding roles.	Clear designation: one name, one responsibility.
“That’w ’s how we’ve always done it this way. ″	A culture of the past.	Making successes visible: the pilot project as proof.
“That’s just bureaucracy.”	A lack of understanding of the benefits.	Examples: What the mesh prevents – and what it enables.

“Resistance isn’t an obstacle. It’s a signal. It shows thw r the organisation is not yet ready – andw where to start."

The success factors

The introduction of the mesh is not a sure-fire success. It requires clear success factors.

Factor	Significance
Leadership	Without the backing of senior management, the mesh will not be introduced.
Clear roles	Every node needs a person in charge – no role, no team.
Modularity	One node at a time – no 'Big Bang' approach.
Visibility	Successes must be made visible – both internally and externally.
Continuity	The Mesh is not a project – it is a process.

"Success is no accident. It is the result of clarity, discipline and leadership."

The transition

Practice shows that the Accountability Mesh is not a theoretical construct. It is a workable architecture. But it is not a sure-fire success either. It requires leadership, clarity and continuity.

And it requires a vision. A vision of what the future of AI governance looks like – and why the Mesh is the only way to shape it.

I describe this vision in the final chapter.

Summary of Chapter 5

Section	Key message
The basic rule	Modularity rather than a ‘Big Bang’.
Stage 1	The pilot: one system, one hub, one person in charge.
Stage 2	Scaling: multiple systems, multiple nodes.
Stage 3	The conglomerate: federated responsibility.
Stage 4	Continuous improvement.
The obstacles	And how to overcome them.
The success factors	Leadership, clear roles, modularity, visibility, continuity.
The transition	The vision: What the future of AI governance looks like.

Chapter 6: The Vision

What the future of AI governance looks like

We have described a problem. We have outlined a solution. We have shown a way to implement it.

But what does this mean for the future?

Not for the future ten years from now. But for the future that has already begun. Systems are becoming more complex. Providers are becoming more powerful. Interdependencies are growing. And vanity is not going to disappear.

The question is not whether complexity is increasing. The question is whether we can manage it.

The answer is: yes – if we build hubs. If we federate responsibility. If we introduce the Accountability Mesh.

“The future is not whaw e happens. It is whw ew e build. Andw e can build it build it –w whenw wew want it to be.”

The three scenarios

There are three possible scenarios for the future of AI governance. And each scenario has consequences.

Scenario	Description	Consequence
Scenario 1: The Monolith	One provider dominates the market. All systems come from a single source.	Maximum efficiency. Minimum autonomy.

Scenario 2: Chaos	Many providers, many systems, no control.	Maximum flexibility. Minimum security.
Scenario 3: The Mesh	Many providers, many systems - but with nodes and federated responsibility.	Maximum autonomy. Manageable complexity.

“The future is not a choice betwew e efficiency and security. It is a choice betwew e control and loss of control.”

Scenario 1: The Monolith

One provider dominates the market. All systems come from a single source. All agents are part of an ecosystem. All data is stored in one place.

Advantages:

- Maximum efficiency.
- Uniform governance.
- Clear accountability.

Disadvantages:

- Maximum dependence.
- Minimal autonomy.
- No Plan B.

“The monolith is convenient – until it is no longer convenient. And then there is no way outw eg.”

Scenario 2: Chaos

Many providers, many systems, no control. Every department buys what it wants. Every agent does what they want. Every interface is a gap.

Advantages:

- Maximum flexibility.
- Minimal dependence on a single provider.
- Rapid adaptation.

Disadvantages:

- Minimal security.
- No accountability.
- No control.

“Chaos is flexible – until it tips over. And then there’s no holding on.”

Scenario 3: The Mesh

Many providers, many systems – but with hubs and federated responsibility. Every system has a person in charge. Every interface has an owner. Every process has a name.

Advantages:

- Maximum autonomy.
- Manageable complexity.
- Clear accountability.
- Flexibility without chaos.

Disadvantages:

- Requires discipline.
- Requires leadership.
- Requires patience.

“The mesh isn’t easy. But it’s the only way to manage complexity – without losing control.”

The principles of the future

The Mesh is not a technology. It is a mindset. And this mindset is based on five principles.

Principle	Meaning
1. Federated Responsibility	Responsibility is not centralised - it is distributed. It is always clearly assigned.
2. Deterministic control	Control is not probabilistic - it is deterministic. Code stops code.
3. Transparency through auditing	Every handover is documented. Every decision is traceable.
4. Modularity rather than a 'Big Bang'	The mesh grows in stages – not all at once.
5. Leadership rather than administration	The Mesh needs leadership – not just management

"The future is not the result of technology. It is the result of a mindset."

The role of people

In a world of multi-layer conglomerates, the role of people is not diminishing – it is becoming more important.

Role	Task
System owner	Responsibility for the entire system.
Interface Owner	Responsibility for the handover between systems.

Process Owner	Responsibility for the overall result.
Human in the loop	Humans as the final resort – in exceptional cases and when situations escalate.

“Humans are not the schw achstelle. They are the last line of defence. And they are the ones who bear verantw ortung – not the machine.”

Europe’s role

The future of AI governance will not be decided in Silicon Valley alone. It will also be decided in Europe – if Europe is prepared to shape it.

Opportunity	Significance
European sovereignty	Own models, own infrastructure, own regulations
multi-vendor architectures	No dependence on a single provider or country.
Accountability Mesh	Federated responsibility – as a European model

“Europe can shape the future of AI governance –w – if it is prepared to take on responsibilityw . Not as a regulator. As an architect.”

The Call

The future is not set in stone. It is being shaped – by those who act today.

- By leaders who take responsibility.

- By architects who build hubs.
- By companies that choose sovereignty.
- By a Europe that has a voice of its own.

“The future is not whw ’s happening. It is whw ’sw ’re doing. So let’w do it.”

The conclusion

This add-on ends here. But the work is only just beginning.

The Accountability Mesh is not a finished product. It is an architecture. An architecture for a world in which systems are becoming more complex – and accountability is becoming clearer.

It is the logical continuation of HB4L, Safe in the Swarm and Multivendor Agentic Swarms. It is the next step.

“For a long time, I believed that technology was the problem. But I was wrong. Technology is merely the tool. The problem is vanity. And the solution is Verant’w ortation. Node by node. Name by name. Person by person.”

Summary of Chapter 6

Section	Key message
The three scenarios	Monolith, Chaos, Mesh – and why the Mesh is the only manageable future.
The five principles	Federated responsibility, deterministic control, transparency, modularity, leadership.
The role of humans	People are not becoming less important – they are becoming more important.

The role of Europe	Europe can shape the future – if it takes responsibility.
The call	The future is what we do. So let’s do it.
The conclusion	The Mesh is not a technology. It is a mindset.